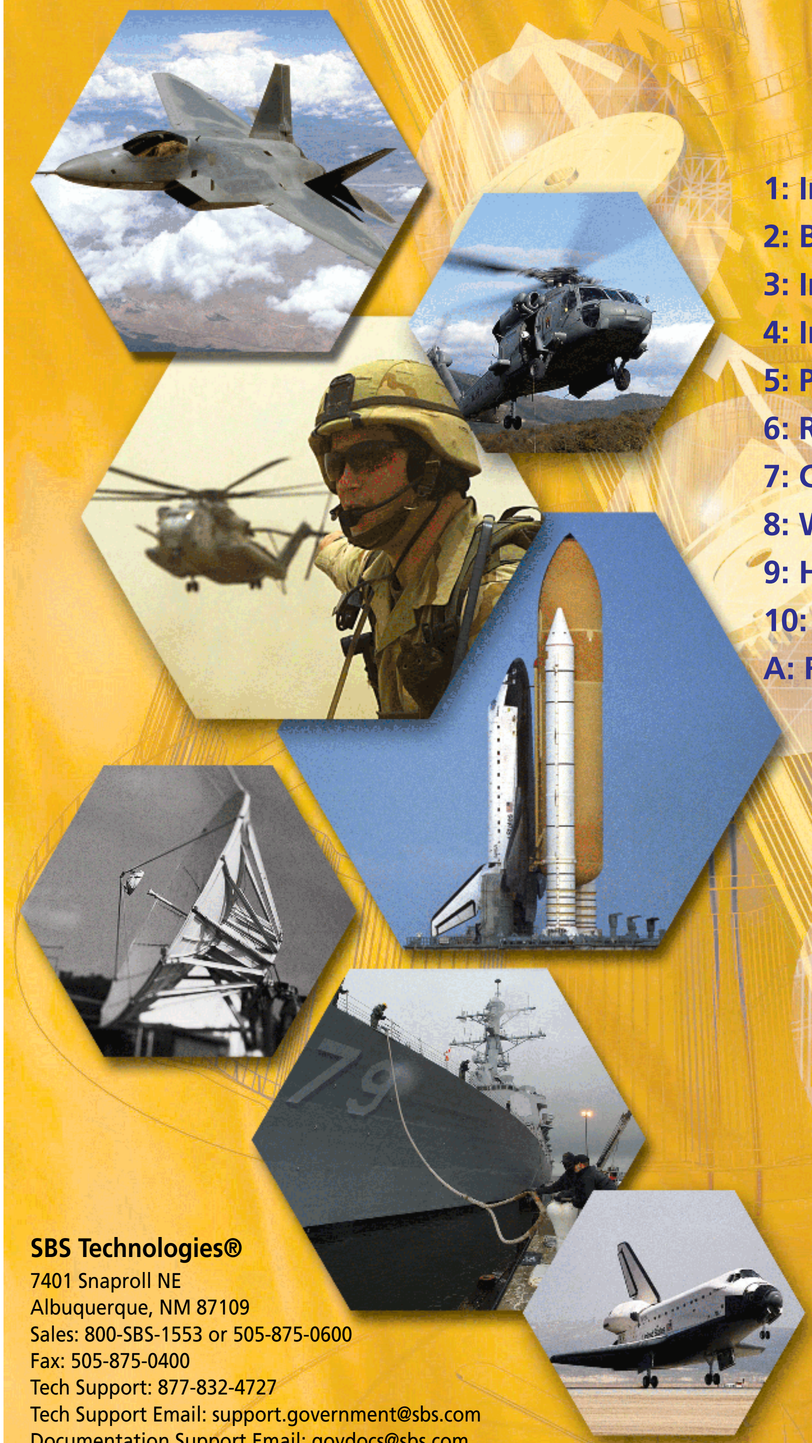
Getting Started with the ABI/ASF-V6-1 or ABI/ASF-V6-2



SBS Technologies®
7401 Snaproll NE
Albuquerque, NM 87109
Sales: 800-SBS-1553 or 505-875-0600
Fax: 505-875-0400
Tech Support: 877-832-4727
Tech Support Email: support.government@sbs.com
Documentation Support Email: govdocs@sbs.com
http://www.sbs.com
http://www.resource.sbs.com

Applies to models:
ABI-V6-1, ASF-V6-1,
ABI-V6-2, ASF-V6-2

Doc PN: 504-553045-00

SBS Technologies®

Getting Started with the ABI/ASF-V6-1 or ABI/ASF-V6-2

SBS Technologies, Inc., 7401 Snaproll NE, Albuquerque, NM 87109

1: Introduction

This chapter introduces the *Getting Started with the ABI/ASF-V6-1 or ABI/ASF-V6-2* manual. It describes the contents of this manual, and the terminology and conventions used in this manual. The sections are as follows:

- ➢ Contents of Getting Started with the ABI/ASF-V6-1 or ABI/ASF-V6-2
- ➢ Contents of MIL-STD-1553 ABI/ASF User's Manual
- ➢ Contents of the Integrated Avionics Library Reference Manual
- ➢ Terminology
- ➢ Conventions

Cross Reference: Appendix A gives a brief summary of technical revisions made to this manual.

1.1 Contents of *Getting Started with the ABI/ASF-V6-1 or ABI/ASF-V6-2*

This manual applies to the following model numbers (the last digit in the model number indicates the number of channels):

- ➢ ABI-V6-1
- ➢ ASF-V6-1
- ➢ ABI-V6-2
- ➢ ASF-V6-2

The intention of this manual is to assist you in getting the ABI-V6-1, ASF-V6-1, ABI-V6-2, or ASF-V6-2 up and running as quickly as possible. It addresses the following:

- ➢ Installing the hardware
- ➢ Installing the software
- ➢ Configuring the software for your operating system
- ➢ Testing
- ➢ Basic module operation

This manual assumes that you will be using the SBS Integrated Avionics Library to operate the module. See Section 10.5 for instructions on starting up the module without using the library.

1.2 Contents of *MIL-STD-1553 ABI/ASF User's Manual*

The accompanying *MIL-STD-1553 ABI/ASF User's Manual* contains complete details on module programming and operation, including the following:

- MIL-STD-1553 programming and structures (Chapters 4–15)
- Sample 1553 applications

1.3 Contents of the *Integrated Avionics Library Reference Manual*

The *Integrated Avionics Library Reference Manual* manual provides information on using the included avionics libraries in your own application.

1.4 Terminology

Table 1.4.1 defines some of the basic terms used throughout this manual.

Table 1.4.1: Basic Terminology

Term	Meaning
BC	Bus controller
BM	Bus monitor
Bus	A single 1553 bus connection (i.e., Bus A or Bus B)
Channel	One complete, dual-redundant 1553 bus interface.
Device	A logical entity that corresponds one-for-one with a 1553 channel and a device entry in the *sbs_dev.cfg* configuration file
Dual-redundant	Includes both a primary and a secondary connection (i.e., Bus A and Bus B make up a dual-redundant bus)
Firmware	Program running in the ABI/ASF digital signal processors that controls all 1553 operations. The firmware must be loaded upon device initialization.
RT	Remote terminal
SA	Subaddress
Word	A 16-bit value; i.e., two bytes

1.5 Conventions

The following conventions appear in this document. These conventions may differ from those used in other SBS publications. The subsections listed below describe each convention in more detail:

- Typographic Conventions
- Words Having Special Meaning
- Compound Keystrokes and Menu Selections
- Symbols

1.5.1 Typographic Conventions

Table 1.5.1 shows the typographic conventions used in this document.

Table 1.5.1: Typographic Conventions

Element	Use in Body Text	Use in Procedures
Italic	➢ Cross references to other SBS publications ➢ Filenames and directory paths ➢ Emphasis	➢ Cross references to other SBS publications ➢ Filenames and directory paths
Bold	➢ (Not used in body text)	➢ Controls, dialogs, menus, and text or numeric fields that appear on the screen ➢ Keys on your keyboard
`Courier Roman`	➢ Code examples	➢ Simulating the appearance of screens
`Courier Bold`	➢ Library function calls and syntax ➢ Emphasizing lines of code	➢ Commands and other information that you type as given
Angle brackets, e.g.,< >	➢ Enclosing variable information that you type (without the brackets) in place of a dummy variable	➢ Enclosing variable information that you type (without the brackets) in place of a dummy variable

The point size of the text varies depending on whether it is used in body text, code examples, notes, screens, or procedures.

1.5.2 Words Having Special Meaning

In procedures, the words “Enter” (or “enter”) and “Type” (or “type”) have special meanings that are indicated in Table 1.5.2.

Table 1.5.2: Words with Special Meaning

Word	Meaning
Enter	Key in the specified text or variable information and press the **Return** key.
Type	Key in the specified text. Do not press **Return**.

1.5.3 Compound Keystrokes and Menu Selections

Compound Keystrokes Whenever a procedure instructs you to press multiple keys, a double angle bracket "»" separates the names of the keys. Table 1.5.3 shows an example.

Menu Selections Whenever a procedure instructs you to select an item from a pull-down menu, a double angle bracket "»"separates the menu items. Table 1.5.3 shows an example.

Table 1.5.3: Examples of Notation for Compound Keystrokes and Menu Selections

Instruction	Meaning
Press **Ctrl** » **Alt** » **Delete**.	Press the **Ctrl**, **Alt**, and **Delete** keys simultaneously.
Select **File** » **Open**.	Select **Open** from the **File** menu.

1.5.4 Symbols

The following symbols appear throughout this manual:

Warning: Paragraphs next to this symbol contain information critical to module operation or to your safety.

Note: Paragraphs next to this symbol contain information important to module operation.

Tip: Paragraphs next to this symbol contain useful tips.

Cross Reference: Paragraphs next to this symbol contain cross references to other parts of this manual, or to other SBS publications.

Software Cross Reference: Paragraphs next to this symbol contain cross references to software media included with this product.

2: Before You Begin

The sections in this chapter describe what to do after receiving and prior to installing your card. The sections are the following:

- What You Should Have Received
- Unpacking the Card
- What You Will Need

2.1 What You Should Have Received

For a Single-Device V6 (ABI/ASF-V6-1)

- ABI-V6-1 or ASF-V6-1 interface module
- Cable assembly

For a Dual-Device V6 (ABI/ASF-V6-2)

- ABI-V6-2 or ASF-V6-2 interface module
- Two cable assemblies

For All Module Types

- SBS Resource CD - Contains PDF versions of this manual, the *MIL-STD-1553 ABI/ASF User's Manual,* and the *Integrated Avionics Library Reference Manual*

2.2 Unpacking the Card

Warning: This is an electronic product that is sensitive to electrostatic discharge. Take normal precautions in handling the card to prevent damage.

- Carefully unpack the card and inspect it for physical damage that might have occurred during shipping.
- If you have a damaged card, contact the SBS technical support group that handles maintenance, repairs, and warranties in Albuquerque. When you call us, give us the serial number of your card, and have the card available in case we have questions about its condition.

2.3 What You Will Need

The ABI/ASF-V6 product package includes all items required to operate the card on your chassis except for the following:

For a Single-Device V6

- Two MIL-STD-1553 bus terminators, to perform a bus test on the module
- Two single bus couplers or other appropriate transformer coupling devices, to connect to an actual 1553 bus

For a Dual-Device V6

- Four MIL-STD-1553 bus terminators, to perform a bus test on the module
- Two dual or four single bus couplers or other appropriate transformer coupling devices, to connect to an actual 1553 bus

Cross Reference: See Subsection 9.2.8 for ordering information and part numbers for these items.

3: Installing the Software

The sections in this chapter include descriptions of the provided software disks and instructions on software installation. This chapter contains the following sections:

- Support Software
- Copying the Software to Your Host System

3.1 Support Software

SBS provides support software for its MIL-STD-1553 products as part of the Integrated Avionics Library on the following media:

- SBS Resource CD

SBS Resource CD

The SBS Resource CD contains the following:

- Integrated Avionics Library, including C library source files, DLLs, sample applications, and the console mode version of the Unit Test executable
- Device drivers necessary to support the interface between the libraries and your computer system
- SBS PASS demo software
- Product documentation in PDF format (requires Adobe Acrobat Reader)
- Firmware files that have to be downloaded to the V6 card upon initialization.

3.2 Copying the Software to Your Host System

Use the following instructions to copy the software to the system in which you are installing the V6 card. The following two sections include separate instructions for PC systems and for non-PC systems:

- Windows Operating Systems
- UNIX-Based Operating Systems

3.2.1 Windows Operating Systems

Unless otherwise specified, the installation batch files will place the Integrated Avionics Library on your system. The *Integrated Avionics Library Reference Manual* provides information on using the library in your own application, and operating system dependent software in the *c:\sbs_ver<x.yz>* folder (directory) in your Windows 95/98/ME/2000 or Windows NT environment, where *<x.yz>* is the version number of the current release.

1. Install the library by completing the following steps:

   - Insert the SBS Resource CD into your CD-ROM drive.
   - Start the Explorer and navigate to the CD.
   - Double-click on the Library folder.
   - Double-click **Setup.exe** from the Explorer.
   - Follow the instructions that appear on the screen.

   If you select all of the defaults, it creates the directory structure shown in Figure 3.2.1.

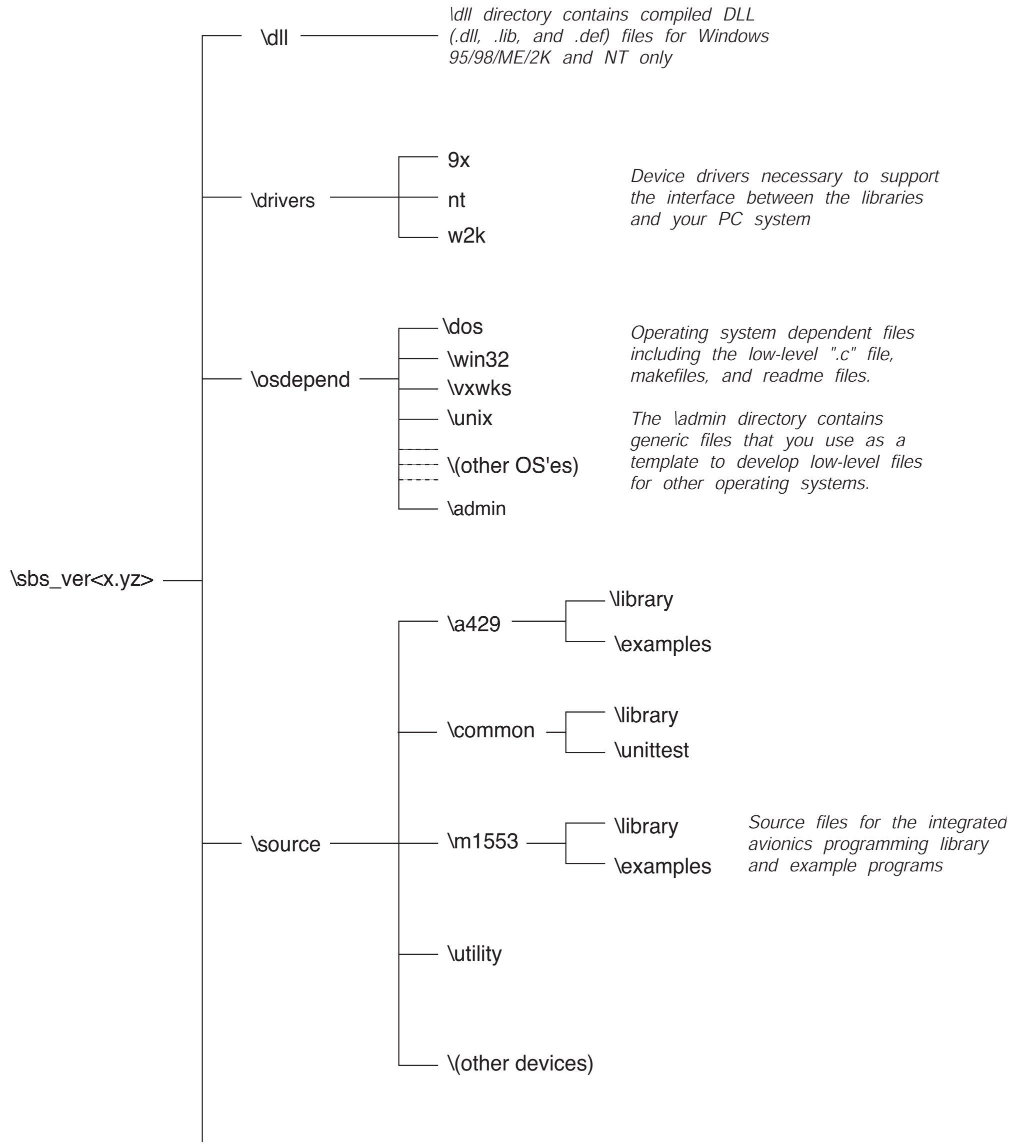


Figure 3.2.1: Default Directory Structure

Note: If your default directory structure differs from that shown, please contact SBS for further instructions.

2. Create the SBS user directory structure shown in Figure 3.2.2 by adding *working* and *firmware* directories under the *sbs_ver<x.yz>* directory, where *<x.yz>* is the version number of the current release.

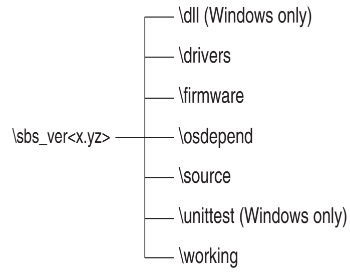


Figure 3.2.2: User Directory Structure

Note: The *\working* directory is a place for you to create and work with files without disturbing other files and directories.

3. Copy the firmware file(s) directly from the Firmware disk to the *\sbs_ver<x.yz>\firmware* directory on your system.

3.2.2 UNIX-Based Operating Systems

To install the interface libraries and operating system software on your UNIX-based system, you must first copy the software to a PC system on your network, then transfer it via ftp to the UNIX-based system in which you are installing the V6 card. The steps below guide you through the process.

1. Copy the software to a PC system on your network by following procedure steps 1–3 beginning on page 8 in the preceding subsection, *Windows Operating Systems*. Make sure to select the files needed for your target UNIX-based system, not the PC system.

2. Enter **`ftp`** to access your host system.

3. Create the directory structure shown in Figure 3.2.3 using the ftp **`mkdir`** utility:

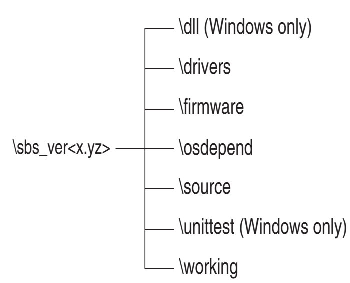


Figure 3.2.3: User Directory Structure

4. Use the ftp **`put`** or **`mput`** utility to transfer the interface library files from the PC environment to the host system (into the above tree structure).

Note: If you have a V6 card with the IRIG option, your Firmware disk should contain two subdirectories, *decode* and *hostload*, each of which contains a different firmware load.

1. Copy the firmware file from the *decode* subdirectory if you will be using an external IRIG signal. This firmware load will not allow you to preset the time (preset time values will be overridden by the external IRIG time).

2. Copy the firmware file from the *hostload* subdirectory if you want to generate the IRIG signal internally from a preset time value. This firmware load will not accept an external IRIG signal.

4: Installing the Hardware

The sections in this chapter provide instructions on how to configure and install the ABI/ASF-V6-1 and ABI/ASF-V6-2 hardware. The sections are as follows:

- ➢ Determining the Base Memory Address, Address Mode, Interrupt Vector, and Interrupt Level
- ➢ Installing the Card
- ➢ Making Auxiliary Connections for External Signals
- ➢ Attaching the Cable Assemblies

4.1 Determining the Base Memory Address, Address Mode, Interrupt Vector, and Interrupt Level

Prior to installing the ABI/ASF card, it is important to select the base memory address, address mode, interrupt vector, and the interrupt level. Table 4.1.1 presents the recommended values for these parameters. For more information on system resources, contact your system administrator or consult the documentation delivered with your computer system.

Table 4.1.1: Recommended Parameter Values

Parameter	Recommended Default Settings
Base Memory Address	48000000h
Address Mode	A32
Interrupt Vector	F0h
Interrupt Level	7

4.1.1 Base Memory Address

In the host VMEbus system, locate an unused region of memory of the appropriate size as shown in Table 4.1.2.

Table 4.1.2: Memory Size for the V6

Model	Memory Size (Bytes)
Single-Device V6	512 k or 80000 hex
Dual-Device V6	512 k or 80000 hex

Figure 4.1.1 presents a memory layout for a Motorola 680x0 processor. As shown, the processor address (local address of VMEbus address 0) starts at 0000h for A32 addressing.

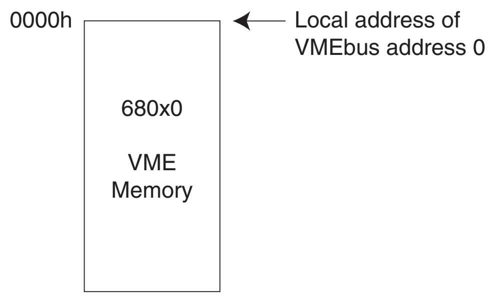


Figure 4.1.1: Motorola 680x0 Memory Layout

Figure 4.1.2 presents a VxWorks system memory layout for a Power PC 604 processor. As shown, the processor address (local address of VMEbus address 0) starts at D0000000h for A32 addressing.

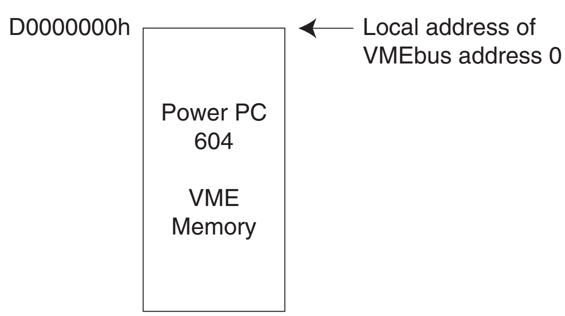


Figure 4.1.2: VxWorks System Memory Layout for a Power PC 604

Note: Please see the documentation supplied with the Processor being used (i.e., Motorola 680x0, Power PC, SPARC, etc.) to determine the processor address (local address of VMEbus address 0).

The location for VME memory region varies depending on the CPU. List the processor address (local address of VMEbus address 0) for your system in the space provided below:

Processor Address: ________________

You must configure the beginning address of the selected region as the base memory address of the V6. List this address in the space provided below:

Selected Base Memory Address: ________________

4.1.2 Address Mode

The address mode reflects the number of address lines used by your system. The V6 card supports both A32 and A24 VMEbus address modes. It supports both supervisor and non-privileged modes. Determine the desired address mode for your system, and list it in the space provided below (A32 is the most common and is the factory default setting for the V6).

Selected Address Mode: ________________

4.1.3 Interrupt Vector and Interrupt Level

Valid values for the interrupt vector are in the range of 00h to FFh. Valid values for the VME interrupt level are in the range of 0 to 7.

Selected Interrupt Vector: ________________

Selected Interrupt Level: ________________

4.2 Installing the Card

This section describes the procedures for installing the ABI/ASF-V6-1 or ABI/ASF-V6-2 card and verifying that the card is properly memory mapped. The topics are as follows:

- Installing the ABI/ASF-V6-1 or ABI/ASF-V6-2 Card
- Verify that the Card is Properly Memory Mapped

4.2.1 Installing the ABI/ASF-V6-1 or ABI/ASF-V6-2 Card

Note: It is necessary to set the Base Memory Address and the Address Mode prior to installing the card. To set the Base I/O Address, refer to Setting the Switches on page 63.

After the switches are properly set for the base memory address and address mode, do the following:

1. Install the V6 module in the host computer in accordance with the system manufacturer's instructions.
2. Locate the LEDs on the front panel of the V6, as shown in Figure 4.2.1.
3. Power up the host system and verify the following *for each V6 channel*:

LEDs

- The RUN LED should be red.
- The Failure Indicator (CH) LED should be off.

If the LEDs are not in the states described above, contact SBS Technical Support.

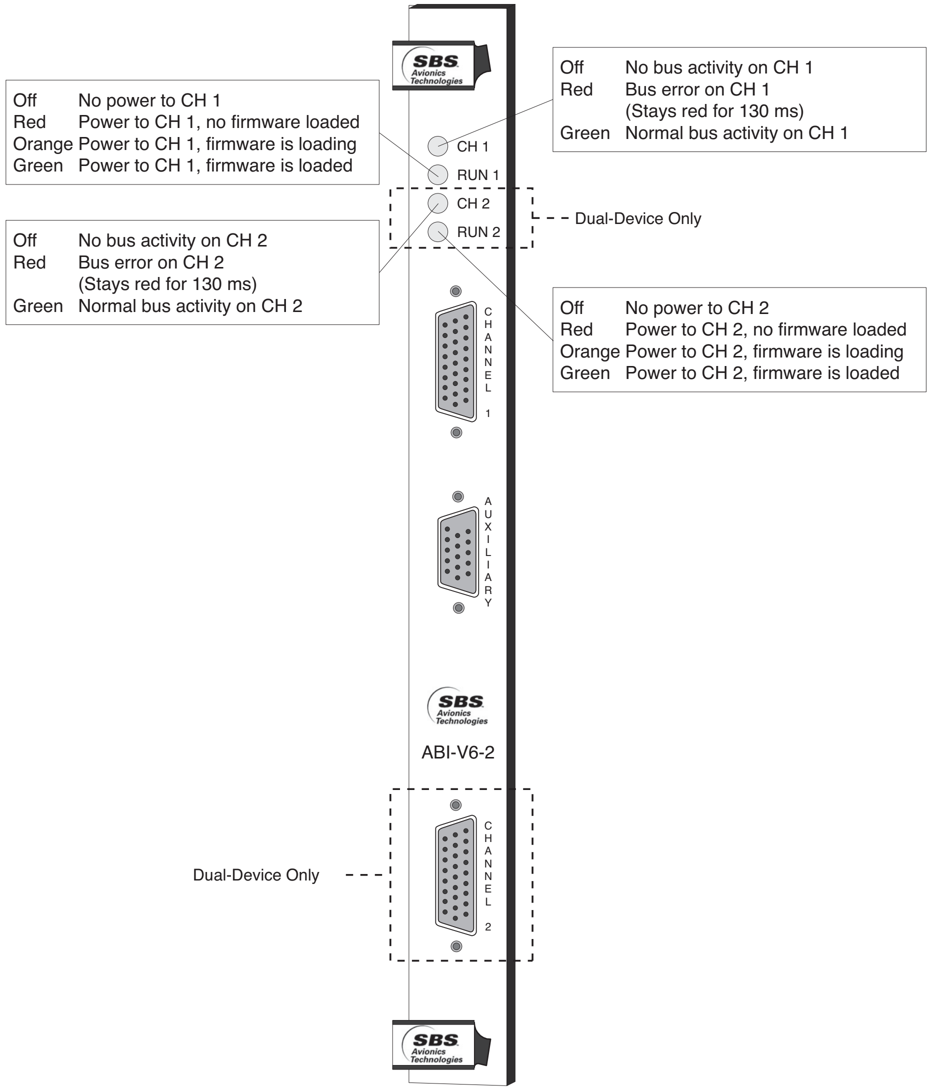


Figure 4.2.1: V6 Front Panel LEDs

4.2.2 Verify that the Card is Properly Memory Mapped

If the operating system you are using has a "read from" or "write to" (peek and poke) memory utility, you can ensure the correct processor address and base address have been specified by performing the steps below:

1. Write `FFFF` to the Channel 1 CSR which is located at the sum of the processor address and the base address.

   The **RUN** LED for Channel 1 should turn green. If the LED does not turn green, verify that the processor address is correct and that the base address is set properly on dip switches SW1 and SW3.

2. Write `0000` to the Channel 1 CSR.

   The **RUN** LED for Channel 1 should turn off.

3. Write `0001` to the Channel 1 CSR.

   If the LED turns green, it means that the operating system does not swap bytes. Skip to Step 5.

4. Write `0100` to the Channel 1 CSR.

   If the LED turns green, it means that the operating system swaps bytes. If you're using the ABI/ASF Programming Libraries, you need to set the compiler directive **`#define BITFIELD_1`** in the *sbs_sys.h* file.

5. Verify that Channel 2 is operational by writing either `0001` or `0100` (whichever is correct for your operating system) to the Channel 2 CSR which is located at the sum of the processor address, base address, and 20000h.

   The **RUN** LED for channel 2 should turn orange.

4.3 Making Auxiliary Connections for External Signals

If you are planning to use external signals, you need to make the proper connections to the Auxiliary connector on the front panel shown in Figure 4.2.1. See Table 9.2.4 on page 65 and Figure 9.2.6 on page 68 for the pinout locations. The remaining topics in this section describe making the connections for IRIG signals and for external triggers.

IRIG Signal If you are using an external IRIG signal as your timing source, make the following connections:

1. Connect the IRIG signal to pin 1 (IRIG) of the Auxiliary connector on the V6 front panel.

Note: The IRIG input impedance is 10 kΩ.

2. Connect the ground to pin 2 (GND) of the Auxiliary connector on the V6 front panel.

Cross Reference: For more information on IRIG signals, see the Device Management Programing Chapter of the *ABI/ASF User's Manual*.

External Trigger If you are using an external trigger (in or out), make the following connections:

1. Connect the trigger signal to pin 8 of the Auxiliary connector on the V6 front panel.
2. Connect the ground to pin 9 (GND) of the Auxiliary connector on the V6 front panel.

Cross Reference: For more information on external triggers, see the Device Management Programing Chapter of the *ABI/ASF User's Manual*.

4.4 Attaching the Cable Assemblies

The V6 module requires that you use the included cable assembly for both testing the card and for actual 1553 operation. The cable assembly provides leads to attach the V6 to the 1553 bus.

1. Attach the cable assembly to the Channel 1 connector on the front panel of the card.

2. If you have a dual-device V6, attach the second cable assembly to the Channel 2 connector on the front panel.

Before Testing the V6

3. Attach 1553 bus terminators to each Bus lead on the cable assemblies), as illustrated in Figure 4.4.1.

Cross Reference: Before operating the V6 on a 1553 Bus, see the instructions on page 53 of this manual for connecting the V6 to a 1553 bus.

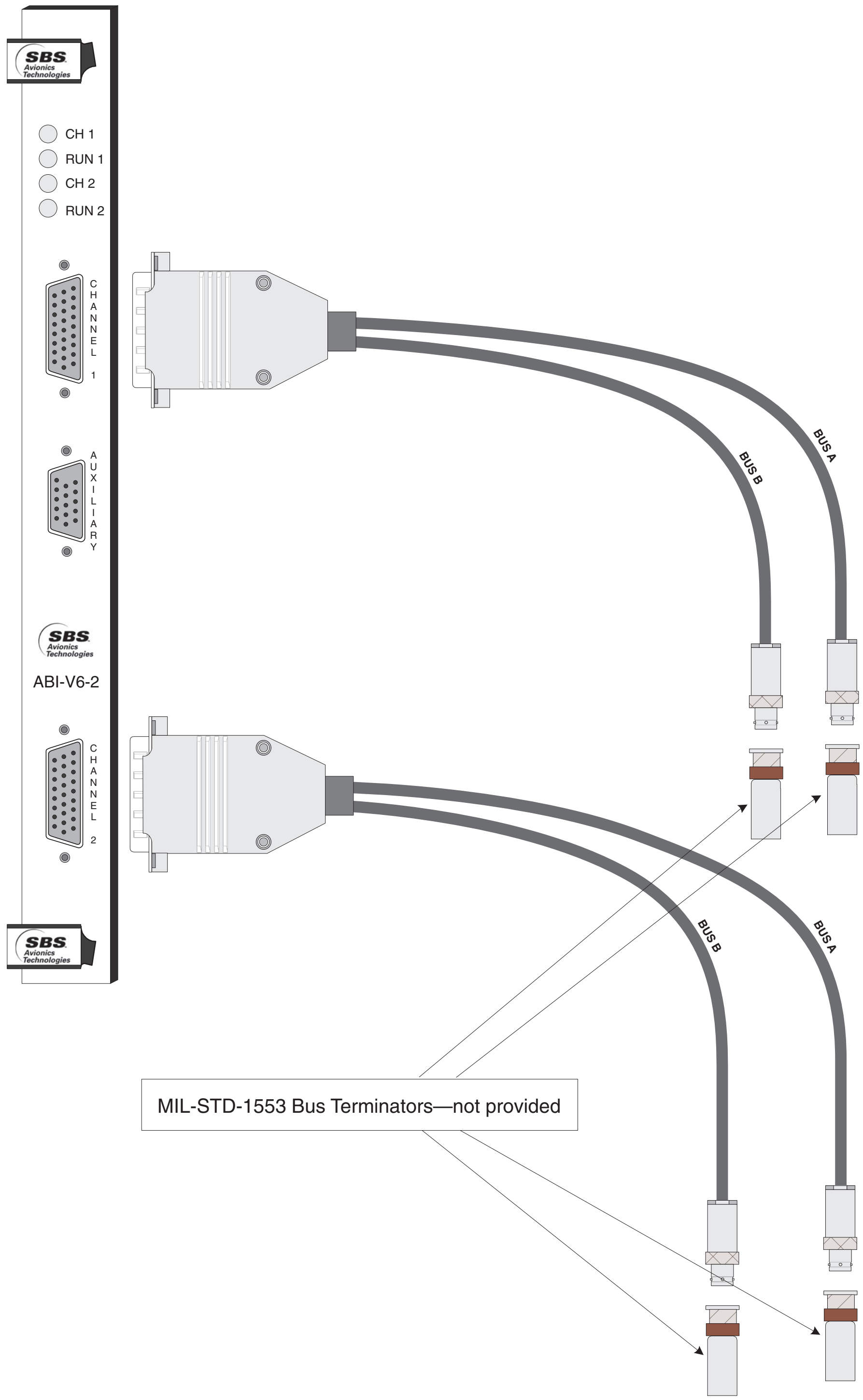


Figure 4.4.1: Attaching the Cable Assembly to the V6

5: Performing System-Specific Installation

The sections in this chapter provide instructions on how to install the files specific to your operating system and platform. This chapter provides instructions for the following systems:

- DEC UNIX
- IRIX
- LYNX
- Solaris
- vxWorks (DEC AXPVME)

Note: If you are using a system not covered in this chapter, see Chapter 21 of the *Integrated Avionics Library Reference Manual* for instructions on installing the files for your system.

5.1 DEC UNIX

This section details the SBS device driver for Digital UNIX Version 3.2c. The device driver also has the framework to work within the Digital UNIX 4.0 (Platinum) operating system.

5.1.1 Memory Map

The device driver maps the SBS device to provide a D16 access path to the V6 memory. It provides access for the full 256 kbytes of memory even though the V6 firmware can access only the lower 128 kbytes. Even though the ABI-V6 has only 256 kbytes of memory, a total of 256 k + 8192 bytes are mapped into the users process. The first 256 kbytes map directly to the memory and control registers on the ABI-V6 board. The top 8192 bytes (size of a page on the Alpha) is a shared kernel buffer used for interrupt notifications.

The shared kernel buffer provides notification wakeup and event counters for various events which could happen on the SBS device. Interrupts notify the device driver of these events. When the device driver receives an interrupt, it reads the interrupt queue and handles all synchronization with the firmware. Depending on what event caused the interrupt, the appropriate count will increment, and wakeups will post on the event counters. User processes issue **`ioctl()`** commands to the driver to wait for specific events.

Of the 8192 bytes mapped into the user space, it actually uses only the first 11 words. These 11 words correspond to the 11 different generated interrupt types. The shared kernel buffer reinitializes during the first opening of the device. It does not reinitialize until the last process closes the device, and a new process reopens the device.

Threads, waiting to be notified of a specific event, use this table to identify the location corresponding to the desired event. For example, if a thread is created which handles interrupt queue overflows, that thread will wait on event #0x7. If the interrupt queue overflows, all threads waiting on event 0x7 awake.

There are constants defined in the *sbs.h* file which correspond to the kernel buffer locations for all events. We strongly recommend that you use the definitions rather than using hard coded values. All definitions relating to the ABI-V6 device have a prefix of **`SBSV6_KBUFFER_`**.

5.1.2 Driver Distribution Format

We distribute the driver as a single compressed tar file. Use the following procedure to extract the contents of the tar file:

1. To uncompress the tar file, use the following command:
   `uncompress sbsdrv302.tar`

Note: It is not necessary to specify the trailing *.z* in the filename specification. The *uncompress* program will replace the compressed tar file with an uncompressed tar file. The contents of the tar file now need to be extracted.

2. To extract the contents of the tar file, execute the following command:
   `tar xvf sbsdrv302.tar`

   - The **`x`** option means to e**x**tract files.
   - The **`v`** option means to be **v**erbose.
   - The **`f`** option says to use the next command line option as the input file.

5.1.3 Installing the Kit

Install the SBS device driver using the standard **`setld`** command. After extracting the contents of the tar file, issue the following command from the same directory that you extracted the tar file into:

`setld -l output`

The following messages and prompts display:

```
*** Enter subset selections ***

The following subsets are mandatory and will be installed automatically un-
less you choose to exit without installing any subsets:

     * SBS Technologies 1553/429 VME Device Driver

You may choose one of the following options:

     1) ALL of the above
     2) CANCEL selections and redisplay menus
     3) EXIT without installing any subsets

Enter your choices or press RETURN to redisplay menus.

Choices (for example, 1 2 4-6): 1

You are installing the following mandatory subsets:
```

```
      SBS Technologies 1553/429 VME Device Driver

You are installing the following optional subsets:

Is this correct? (y/n):  y

SBS Technologies 1553/429 VME Device Driver
   Copying from output (disk)
   Verifying

Configuring "SBS Technologies 1553/429 VME Device Driver" (SBSDRV302)
```

5.1.4 File List

When the kit is installed, the default installation directory */usr/opt* creates the following files:

```
# ls -lR SBSDRV302
total 4
drwxr-xr-x   2 root     system       512 Sep  4 09:57 docs
drwxr-xr-x   2 root     system       512 Sep  4 09:57 driver
drwxr-xr-x   2 root     system       512 Sep  4 09:57 firmware
drwxr-xr-x   2 root     system      1024 Sep  4 09:57 libs

SBSDRV302/docs:
total 2
-rw-r--r--   1 root   system         13 Sep  4 09:44 UsersGuide.ps
-rw-r--r--   1 root   system         13 Sep  4 09:44 UsersGuide.txt
SBSDRV302/driver:
total 98
-r--r--r--   1 root   system       2272 Sep  3 15:23 config.file
-r--r--r--   1 root   system       1640 Sep  3 15:23 files
-r--r--r--   1 root   system       9111 Sep  3 15:23 sbs.h
-rw-r--r--   1 root   system      82548 Sep  4 09:54 sbs.o
-r-xr-xr-x   1 root   system       2124 Sep  3 15:23 stanza.static

SBSDRV302/firmware:
total 368
-rw-r--r--   1 root   system     188480 Sep  3 14:25 f008c.dat
-rw-r--r--   1 root   system     162583 Sep  3 14:25 f009c.dat
```

5.1.5 Driver Source Code

The standard distribution does not include the source code for the SBS device driver. If you have permission to access the driver source code, you will then be given a single **C** source file which needs to be placed in the directory */usr/opt/SBSDRV302/driver*. You also need to edit the file */usr/opt/SBSDRV302/driver/files* and change the word **Binary** to **Notbinary**. The *file* prior to modification is:

```
sbs.c optional sbsvfive sbsvsix device-driver Binary
```

The same file after the required modification to compile a new source is:

```
sbs.c optional sbsvfive sbsvsix device-driver Notbinary
```

After you have changed this file, rebuild the kernel using the following command syntax:

```
cd /usr/sys/config
./config SYSTEMNAME
cd ../SYSTEMNAME
make
```

Note: If you will be making other changes to the driver files, you can defer building the kernel until all changes are made.

5.1.6 Configuration

The information supplied in the device driver configuration file *config.file is* the basis for the SBS driver configuration that is done at system startup time. The device driver configuration file is set up with reasonable defaults, and the default location for the configuration file is */usr/opt/SBSDRV302/config.file*.

5.1.7 Setting the VME Base Address

The default VME base address for the SBS device driver is 0x48000000. This is the same address as the default VME address for the MIL-STD-1553 ABI-V6 board as shipped from SBS. To change the default, edit the drivers *config* file. This filename is */usr/opt/SBSDRV302/config.file*. Edit this file and find the following text:

```
controller sbsvsix0 at vba0 csr 0x48000000 priority 6 vector sbsvsix_intr 0x70
```

Change the csr value from **`0x48000000`** to the desired value. After you change this file, rebuild the kernel using the following command syntax:

```
cd /usr/sys/config
./config SYSTEMNAME
cd ../SYSTEMNAME
make
```

Note: If you will be making other changes to the driver files, you can defer building the kernel until all changes are made.

5.1.8 Setting Interrupt Priority Levels

The default interrupt priority level is six (6). To change the default, edit the drivers *config* file. This filename is */usr/opt/SBSDRV302/config.file*. Edit this file and find the following text:

```
controller sbsvsix0 at vba0 csr 0x48000000 priority 6 vector sbsvsix_intr 0x70
```

Change the text **`priority 6`** to **`priority x`**, where **`x`** is the desired value.

After you change this file, rebuild the kernel using the following command syntax:

```
cd /usr/sys/config
./config SYSTEMNAME
cd ../SYSTEMNAME
make
```

Note: If you will be making other changes to the driver files, you can defer building the kernel until all changes are made.

5.1.9 Setting Interrupt Vectors

The MIL-STD-1553 ABI-V6 device uses a single interrupt vector. The default interrupt vector is 0x70. To change the default, edit the drivers *config* file. This filename is */usr/opt/SBSDRV302/config.file*.

Note: The minimum interrupt vector value on the Alpha Single Board Computer (SBC) is 0x18 (24). Attempts at using interrupt vector values lower than 0x18 will result in the driver not being configured into the kernel. The maximum interrupt vector value is 0xff (255).

Note: The minimum interrupt vector value when using the BIT3 PCI <-> VME adapter is 0x10 (16). Attempts at using interrupt vector values lower than 0x10 will result in the driver not being configured into the kernel. The maximum interrupt vector value is 0xff (255).

Edit */usr/opt/SBSDRV302/config.file* and find the following text:

```
controller sbsvsix0 at vba0 csr 0x48000000 priority 6 vector sbsvsix_intr 0x70
```

Change the text **`0x70`** to the desired value. After you change this file, rebuild the kernel using the following command syntax:

```
cd /usr/sys/config
./config SYSTEMNAME
cd ../SYSTEMNAME
make
```

Note: If you will be making other changes to the driver files, you can defer building the kernel until all changes are made.

5.1.10 Driver Routines

`open()`	Opens a connection to an SBS device for **`read()`**, **`write()`**, or **`ioctl()`** calls
`close()`	Closes the connection to an SBS device
`ioctl()`	I/O Control interface to the SBS devices.
`mmap()`	Memory Map interface for the SBS device

5.2 IRIX

The Silicon Graphics computer systems needs a device driver to map shared memory and handle interrupts. We supply two drivers: one for systems running versions IRIX 4.x and another for systems running versions of IRIX 5.x. This section provides installation procedures to build the drivers for your system.

5.2.1 Driver Distribution Files

The IRIX driver directory (*sbs_ver<x.yz>\osdepend\vme\irix\sbsdrvr*), where <*x.yz*> is the version number of the current release, has the following files:

IRIX 4.x

abi	System file specifying driver information such as the major device number
abi.c	Device driver source code
mem.exm	Mem file example
system.exm	System file example

IRIX 5.x

Makefile	Makefile to build the driver
sbs1553	System file specifying driver information such as the major device number
sbs1553.c	Device driver source code
sbs1553.o	Device driver object file
sbs1553.sm	System file containing hardware information for the device
testabi	Driver test program
testabi.c	Driver test file source code

5.2.2 Driver Installation Procedures

IRIX 4.x The base address for the SBS device depends upon the processor implemented in your Silicon Graphics system. If your system has an “IP4”, “IP6”, or “IP10” processor board, the 32-bit base address is 0xBC000000. If your system has an “IP5”, “IP7”, or “IP9” processor board, use address 0xB3000000.

Note: You may notice that the hardware address does not seem to match the address recognized by the IP processor. When the host processor decodes the address, the prefix B3 or BC is used as an offset to specify an address located in the VMEbus cage reserved for peripheral hardware such as our 1553 board. The processor will then specify a 24 bit VMEbus address by stripping the BC or B3 from the 32 bit address and use the remaining six hexidecimal values.

Note: If you are installing multiple SBS devices, there CANNOT be any address gaps, i.e., the boards MUST be contiguously located.

Follow the steps below to install the device driver into the IRIX 4.x system:

1. Login as root.
2. Execute the command **ls -l /dev | more** and look at the output for the first number (the major device number) associated with each entry. Choose a number which is not being used by another device in your system. We chose 50 on the Personal SILICON GRAPHICS system. This may also be acceptable on other SILICON GRAPHICS systems. Go to the directory containing the driver software: **cd sbs_ver<x.yz>/osdepend/vme/irix**, where *<x.yz>* is the version number of the current release.
3. Execute the following command to copy the *abi* file to its proper location:

   ```
   cp abi /usr/sysgen/master.d/abi
   ```

4. Edit */usr/sysgen/mater.d/abi* and change the field under the heading SOFT to the major device number chosen in step 1. Below is an example *abi* file using 50 as the major device number:

   ```
   *
   * SBS ABI - 1553 interface device
   *
   *FLAG  PREFIX   SOFT   #DEV   DEPENDENCIES
   c      abi      50
   $$$
   ```

5. Execute the following command:

   ```
   mknod /dev/abi c <major#> 0
   ```

   where **<major#>** is the major number chosen in step 1.

6. Edit */usr/sysgen/master.d/mem*. Find the **mmmap_addrs** structure. Near the end of the structure, find the entry **{0, 0}**, which is located at the end of the table delimiter. Before the delimiter entry insert a line that reads:

   ```
   {0x20000, 0x##000000,}
   ```

   where **##** is either **BC** or **B3** depending upon your processor. We supply an example mem file (*mem.exm*) with the distribution disks in the *sbs_ver<x.yz>/ osdepend/vme/irix* directory.

7. Copy */usr/sysgen/system/usr/sysgen/system.b4abi* to the driver directory.
8. Edit */usr/sysgen/system* and go to the line after the last **VECTOR:** entry. Insert the following lines:

   ```
   VECTOR: module=abi vector=0xF0 ipl=7 unit=0 base=0x##000800
   probe=0x##000000 probe_size=2
   ```

   where **##** is either BC or B3 depending upon your processor. This line of information assumes the following:

   - The base address of the ABI or ASF is strapped to 0x000000
   - The ABI or ASF is set for 24 bit addressing
   - Interrupt Level: 7
   - Interrupt Vector: F0

We supply an example system file (*system.exm*) with the distribution disks in the *sbs_ver<x.yz>/osdepend/vme/irix* directory, where *<x.yz>* is the version number of the current release.

9. Compile the driver by entering the following command:

```
cc -c -O -o abi.o abi.c
```

Note: -O, enter upper case "O", not zero. -o, enter lower case "o", not zero.

10. Copy the abi object file to the system boot directory:

```
cp abi.o /usr/sysgen/boot/abi.o
```

11. Save your original kernel in case this new one does not boot:

```
cp /unix /unix.b4abi
```

Warning: DO NOT BYPASS THIS STEP! If you need to reboot under the *unix.b4abi* kernel see the IRIX System Administrator's Guide for more information. The command to reboot under the old kernel prompt should be similar to the following command: **boot dksc(0,1)unix.b4abi**.

12. Change the directory to */usr/sysgen*.

13. Execute the following command:

```
lboot -v
```

14. Watch the messages carefully for the **abi:...** message. This confirms the system found the 1553 board.

15. Execute the following command:

```
cp unix.new  /unix.install
```

16. Execute the following commands to reboot the system:

```
sync;

sync;

sync;

init 0
```

Watch the system messages carefully to verify the new kernel rebooted correctly.

IRIX 5.x The IRIX 5.x driver allows mapping the SBS device memory into the user's address space. It receives interrupts via the *ioctl()* interface (see *testabi.c*).The driver assumes that you use the factory default VMEbus address (0x48000000), IPL (5), and interrupt vector (0xF0) for the first board. If it is necessary to change the base address and interrupt vector on succeeding boards, edit the file *sbs1553.sm* and change these parameters to reflect your configuration.

Follow the steps below to install the device driver into the IRIX 5.x system:

1. Login as root.
2. Enter the following command to make 4 nodes */dev/1553[A-D]*:

   ```
   make node
   ```

3. Compile the driver code by entering:

   ```
   make sbs1553.o
   ```

4. Save the original kernel to ensure that the old version of UNIX kernel is there in case things go wrong.

   ```
   ln /unix /unix.old
   ```

5. Copy everything to its proper place.

   ```
   make install
   ```

6. Make a new UNIX kernel.

   ```
   make newunix
   ```

Note: You should see that the *lboot(1M)* found at least one of your ABI boards in the system. If not, you need to check the VME addresses.

7. Reboot the system by entering the following command:

   ```
   reboot
   ```

8. Test the board by entering the following command:

   ```
   make TestAbi
   ```

5.3 LYNX

LynxOS requires a device driver to handle interrupts and to map virtual memory. We tested this driver with LynxOS version 2.1, 2.3, 2.4.1, and 3.0.1.

5.3.1 Driver Distribution Files

The following files make up the LynxOS driver directory (*sbs_ver<x.yz>\osdepend\lynx)*, where <*x.yz*> is the version number of the current release:

Makefile	Compiles the driver source code in *sbsdrvr.c*, *sbsinfo.c*, and creates a test program from *testdrvr.c*
mk1553dr	Device driver install script
Readme	A text file of these notes
sbsdrvr.c	Device driver source code
sbsdrvr.h	Data definitions used by the device driver
sbsinfo.c	Variable declarations needed when the system is being configured
testdrvr.c	A test program used to test the driver
1553.cfg	A system file containing device information

5.3.2 Driver Installation Procedures

Follow the steps below to install the device driver into the LynxOS system. You must have superuser privileges to install the driver.

1. Edit */sys/lynx.os/CONFIG.TBL* and add the following line at the end of the file:

   ```
   I:1553.cfg
   ```

2. Set the address and interrupt parameters for each board. Edit the *sbsdrvr.h file* and locate the following define statements:

   ```
   #define BA_ADDRx
   #define IO_ADDRx
   #define INT_VECTx
   ```

   where `x` is the device number. Insert the desired base address, I/O address, and interrupt vector values for each of the devices to be installed. The base addresses should be a minimum of 40000h difference between each device.

   Note: If you are installing a dual-device card, give each device the same base address and I/O address.

   Modify the value of `MAXDEVS` to match the number of devices to be installed.

3. All the devices use the same driver but at different nodes. To specify the name of each device node, edit the *1553.cfg* and uncomment one of the following lines for each device.

```
# N:sbs1553a:1:0666

# N:sbs1553b:2:0666

# N:sbs1553c:3:0666

# N:sbs1553d:4:0666

# N:sbs1553e:5:0666

# N:sbs1553f:6:0666

# N:sbs1553g:7:0666

# N:sbs1553h:8:0666

# N:sbs1553i:9:0666

# N:sbs1553j:10:0666

# N:sbs1553k:11:0666
```

a,b,c,d,e, etc. at the end of the device name (**sbs1553x**) maps to 1, 2, 3, 4, etc. of the compile directives (**BA_ADDRx**, **IO_ADDRx**, and **INT_VECTx**) in the *sbsdrvr.h* file. For example, the device name (i.e., **sbs1553a**) corresponds to the device setting directives **BA_ADDR1**, **IO_ADDR1**, and **INT_VECT1** specified in the *sbsdrvr.h* file.

4. If you are installing more than one SBS device or setting the driver up to handle more than one card, edit the file *sbsinfo.c*. An array of structures named **sbsinfo** in *sbsinfo.c* must be initialized for each board in the system. Uncomment **sbsinfo** for each board. Add additional lines if you have more than five boards.

5. Build the kernel by entering the following command:

mk1553dr

6. Enter the following command to make the new UNIX kernel the boot kernel:

MVME162/167 platforms:

makeboot /lynx.os

PowerPC platforms:

See makeboot in the “Utility Program” manual delivered with the Lynx OS distribution or the UNIX man pages.

7. Enter the following command to reboot the system.

reboot -aN

8. Enter **testdrvr** to run the SBS driver test. The output of the test displays on the screen and should look like the following.

```
Opening driver.

  In sbsopen()

  Base addr: xxxxxxxx Level: zz

  Board 0 found.

File Descriptor : y

Mapped Base Address: xxxxxx

  In sbsioctl() - minor dev: 0

1553 Probe Test Pass

Begin WaitForIntr()

  In sbsioctl() - minor dev: 0

  In sbsioctl() - timeout set

T2 done

T1 done

Interrupt Count = 1

Closing driver.

  In sbsclose()
```

The addresses displayed above (*xxxxxx*) corresponds to a virtual address, the file descriptor (*y*) should be an integer, and the interrupt vector (*zz*) should correspond to the vector jumpered on the board. If the test fails, the message **1553 Probe Test Fail** displays. If this occurs, the jumpered address on the board does not match the address in the driver file system. Double-check these values.

Note: To suppress screen messages such as **In sbsopen()** that are generated by the driver, edit the makefile, remove the **-DDEBUG** flag, and recompile the driver.

5.3.3 Hardware Interrupt Operation

You can set up device driver interrupts and handle them by using the **ioctl()** system call. The first parameter to **ioctl()** should be the file descriptor returned from the **open** call. The second parameter must be one of the following request codes:

INTR_INSTALL	Installs the interrupt handler
INTR_WAIT	Waits for an interrupt. When the interrupt occurs, the user's interrupt handler is called.

The third parameter to the **ioctl()** call depends upon the request code. For **INTR_INSTALL**, this parameter is not used, but you must pass a pointer to a character anyway. For **INTR_WAIT**, the third parameter should point to a structure of type **struct intr_entry** as defined in *sbsdrvr.h.*

```
struct intr_entry
{
   void (* handle_1553_intr)();
   int  intr_timeout;
   char *arg;
}
```

The caller must initialize this structure. The element **handle_1553_intr** is a pointer to a function that will be executed with one argument, **arg**, when an interrupt occurs. The **ioctl()** will wait **intr_timeout** hundredths of a second. A value of zero will cause the **ioctl()** to wait indefinitely. Review the procedure in *testdrvr.c* for further details.

5.4 Solaris

The Force Solaris 2.x VMEbus Driver Package compiles using either Sun's Proworks compiler or the gnu compiler. This is a user-level map type driver. We do not implement interrupts, but user-level interrupts and user-level dma are available in the force VMEbus Driver Package.

Note: A big effort in this port was getting the Force Sparc/CPU-20VT to read the Sun Solaris 2.5.1 distribution CD-ROM. The system will not boot from CD-ROM without some "patches" in the OpenBoot NVRAM. Sun's technical support can furnish a workaround for this problem. The patch supplied makes two serial interfaces unavailable (ttyc and ttyd). However, ttya and ttyb are still available.

5.4.1 Driver Distribution Files

The following files make up the Solaris driver directory (*sbs_ver<x.yz>\osdepend\solaris*), where *<x.yz>* is the version number of the current release:

int_vmep.c	VMEplus interrupt support routines
int_test.c	A program that initializes the device to perform BC (and optionally RT functions, and then causes interrupt(s), handles them, and displays the results

5.5 vxWorks (DEC AXPVME)

In this configuration, you compile the applications on the DECUNIX host, then download them to the vxWorks platform for execution.

5.5.1 Driver Distribution Files

The following files make up the standard vxWorks driver directory (*sbs_ver<x.yz>\osdepend\vxworks*), where <*x.yz*> is the version number of the current release:

sbstsk.c	A collection of tasks used to manage the SBS device interrupt handler that interfaces to the vxWorks library
swtypes.h	Data type definitions used by the software routines in the file named *sbstsk.c*

5.5.2 Driver Installation Procedures

1. Modify the software in the files *sbstsk.c* to account for your system configuration.
2. Add **`sbstsk.c`** to the *makefile*.
3. Compile and link the application.

6: Running Unit Test

To verify that the V6 is properly installed and operational, use the instructions in the following sections to run the Unit Test application:

- ➢ Introduction
- ➢ Using the Combined (1553, A429, and WMUX) Unit Test Executable
- ➢ Setting up the Device Configuration File
- ➢ Unit Test Using the Console Mode
- ➢ Troubleshooting

Note: You must have an ANSI compatible terminal or driver in order to run Unit Test.

6.1 Introduction

The *sbs_ver*<*x.yz*>*execs*\\ directory, where <*x.yz*> is the version number of the current release, supplies the executable for Windows for the combined Unit Test application. If you are using one of these operating systems with the V6 card and wish to use the combined Unit Test, proceed to the next section. If you do not have access to the distributed Unit Test executables, if there is not a precompiled executable for your operating system, or if you wish to run a stand-alone 1553 Unit Test, use the instructions in the Compiling Your Application Chapter of the *Integrated Avionics Library Reference Manual* to compile a new Unit Test.

You can begin using the Unit Test by going to Unit Test Using the Console Mode in Section 6.4.

6.2 Using the Combined (1553, A429, and WMUX) Unit Test Executable

This section describes the procedures for using the combined Unit Test. The topics are as follows:

- ➢ Operating Systems with File Systems
- ➢ Operating Systems without File Systems

6.2.1 Operating Systems with File Systems

To use the precompiled, combined Unit Test for Windows, complete the following steps:

1. Copy the Unit Test executable from the *sbs_ver<x.yz>**execs**unittest.exe* directory to the *sbs_ver<x.yz>**working* directory (where *<x.yz>* is the version number of the current release).
2. Copy the *sbs_dev.cfg* file from the *sbs_ver<x.yz>**source**common**library* directory to the *sbs_ver<x.yz>**working* directory.
3. Copy the firmware files from the *sbs_ver<x.yz>**firmware* directory to the *sbs_ver<x.yz>**working* directory.
4. Set up the *sbs_dev.cfg* file as described in the next section, *Setting up the Device Configuration File*.

6.2.2 Operating Systems without File Systems

1. Refer to the Compiling Your Application Chapter of the *Integrated Avionics Library Reference Manual* to determine the necessary files and appropriate compiler directives. Copy the appropriate files into the *sbs_ver<x.yz>**working* directory, where *<x.yz>* is the version number of the current release.
2. Set up the *dev_cfg.h* file as described in the next section, *Setting up the Device Configuration File*.
3. Compile your Unit Test application as described in the Compiling Your Application Chapter of the *Integrated Avionics Library Reference Manual*.

6.3 Setting up the Device Configuration File

You must define the device parameter values for your SBS device(s) in one of the following two files:

- *sbs_dev.cfg*, if your operating system has a file system
- *dev_cfg.h*, if your system does not have a file system

The *sbs_dev.cfg* device configuration file is an ASCII text file containing information that your application uses to initialize one or more SBS devices. The SBS device's initialization procedure calls a parser function (**`sbs_parse_config_file()`**) that reads the information in the configuration file. The parser function must be able to correctly read and verify the information for an SBS device before further library calls can be made. Because the **`sbs_parse_config_file()`** function reads the *sbs_dev.cfg* file at run-time, you can modify *sbs_dev.cfg* without recompiling the application.

If your operating system does not have a file system, you should use the *dev_cfg.h* file instead of the *sbs_dev.cfg* file. The *dev_cfg.h* file is an array of strings with the same format and keywords as the *sbs_dev.cfg* file. It parses in the same manner as the *sbs_dev.cfg* file. However, you must recompile your application each time you edit *dev_cfg.h* for your changes to take effect.

Software Reference: The location of *sbs_dev.cfg* and *dev_cfg.h* configuration files are in the directory in which you installed the Integrated Avionics Library under the *\install\library\source\common\library* directory.

The following sections provide descriptions and examples of the *sbs_dev.cfg* and *dev_cfg.h* files:

- Format
- Keywords
- sbs_dev.cfg File
- dev_cfg.h File

6.3.1 Format

The format of the *sbs_dev.cfg* and *dev_cfg.h* files resembles an *.ini* file found on most PCs. Values contained in this file vary based on your SBS device(s), system hardware, and operating system. Every device requires a separate set of keyword entries in the configuration file. (Thus, each single-device card requires a single set of keyword entries, and each multidevice card requires multiple sets.) The file is set up as follows:

- An equal sign (**=**) links each keyword with a value.
- The variable **<num>** represents a numerical value.
- The variable **<name>** represents a case-sensitive character string.
- The parser recognizes the following characters appended to the value string:
  - **b** (binary)
  - **o** (octal)
  - **h** (hex)
  - **d** (decimal; the default if no character is present)
- A semicolon at the beginning of a line denotes a comment line, and the parser ignores the entire line.

6.3.2 Keywords

Table 6.3.1 describes the keywords required in the *sbs_dev.cfg* or *dev_cfg.h* file for configuring the V6 card.

Table 6.3.1: Required Configuration File Keywords for the V6

Keyword=<Specifier>	Required for:	Description
[DEVICE=<num>]	All operating systems and platforms	This line is required before all other keywords for a device. <num> represents a unique number, starting at 1, that identifies the SBS device.
base_address=<num>	All operating systems and platforms	<num> specifies (in bytes) the location of the SBS device in physical address space. This location is system and device specific. Consult your system user's manual for available memory locations. **Cross Reference:** See the subsection *Base Memory Address* on page 14 for the value you selected.
dd_name=<name>	Windows NT and some UNIX systems (not required for vxWorks)	<name> specifies the UNIX or Windows NT device driver name. It contains a text string with the name of the device driver node associated with the SBS device. For UNIX, <name> is the filename of the driver the */dev* directory. For IRIX 6.4/6.5 systems, <name> is the XIO pathname that points to the memory space of the PCI bus for your card. For Windows NT, this value must correspond exactly to the name of the instantiated driver for the device, which can be found in the device directory using the Device Manager.
device_type=<name>	All operating systems and platforms	<name> specifies the type of SBS card being used. For a single-device V6, set the value to M1553_V6_1. For a dual-device V6, set the value to M1553_V6_1 for the first device and M1553_V6_2 for the second device.

Keyword=<Specifier>	Required for:	Description
`firmware=<name>`	All operating systems and platforms	`<name>` specifies the firmware filename. If you define `NO_FILE_SYSTEM` in *sbs_sys.h*, the firmware will be loaded from a data array found in *firmware.h*. The array is specified by the "`firmware=`" entry in the *dev_cfg.h* configuration file. You can generate the *firmware.h* file using the utility program *setup_fw*.c. Devices that have flash memory still require this field to reprogram the flash memory. **Note:** Verify that the firmware filename listed in the *sbs_dev.cfg* or *dev_cfg.h* file is the same as that of the firmware file provided with the software distribution.
`int_vector=<num>`	All operating systems and platforms	This keyword is used for VMEbus devices only. `<num>` specifies the VME interrupt vector for the device. Valid values are in the range 0 to FFh. **Cross Reference:** See the subsection *Interrupt Vector and Interrupt Level* on page 15 for the value you selected.
`irq_level=<num>`	All operating systems and platforms	This keyword is used for VMEbus and PC devices. `<num>` specifies the interrupt request level for the device. For VMEbus devices, the valid range is 0 to 7. For PC devices, the valid range is 0 to 0FhFor both VMEbus and PC devices, a value of 0 indicates that no IRQ level is selected. **Cross Reference:** See the subsection *Interrupt Vector and Interrupt Level* on page 15 for the value you selected.

6.3.3 *sbs_dev.cfg* File

Note: If your operating system does not have a file system, you must use the *dev_cfg.h* file instead of *sbs_dev.cfg*. See page 43 for information on *dev_cfg.h*.

If your operating system has a file system, define the device parameter values for each SBS device in the *sbs_dev.cfg* file. Modify *sbs_dev.cfg* as follows:

1. Change directories to the *\sbs_ver<x.yz>\working* directory, where *<x.yz>* is the version number of the current release, if you are not already there.
2. Edit the *sbs_dev.cfg* file and verify that the settings are correct for the device type, firmware file(s), and number of devices you are using.
3. Save the file.

Examples of the parameters used for the *sbs_dev.cfg* file appear on the following page for a dual-device ABI-V6 card running under the following systems:

- vxWorks
- LYNX, Solaris, IRIX, or DEC UNIX

These examples show only the required parameters. In the actual *sbs_dev.cfg* file you would remove the comment (semicolon preceding the parameter). All remaining lines are left commented out (preceded by a semicolon). If you are using a single-device card, you should uncomment keywords only for device 1. If you are running your card under an operating system other than those in the examples, refer to Table 6.3.1 to determine the keywords that your operating system requires.

Cross Reference: For an example of the complete *sbs_dev.cfg* file, refer to the *Integrated Avionics Library Reference Manual*.

Note: The values required for your system, including the firmware filename, may differ from those shown below.

vxWorks

```
[DEVICE=1]
base_address=48000000h
device_type=M1553_V6_1
firmware=f005s.dat
int_vector=F0h
irq_level=1
[DEVICE=2]
base_address=48040000h
device_type=M1553_V6_2
firmware=f005s.dat
int_vector=F1h
irq_level=2
```

LYNX, Solaris, IRIX, or DEC UNIX

```
[DEVICE=1]
base_address=48000000h
device_type=M1553_V6_1
dd_name=Your_Device_Path_Here
firmware=f005s.dat
int_vector=F0h
irq_level=1
[DEVICE=2]
base_address=48040000h
device_type=M1553_V6_2
dd_name=Your_Device_Path_Here
firmware=f005s.dat
int_vector=F1h
irq_level=2
```

6.3.4 *dev_cfg.h* File

Note: If your operating system has a file system, you should use the *sbs_dev.cfg* file instead of *dev_cfg.h*. See page 41 for information on *sbs_dev.cfg*.

This file contains the device information to be used in lieu of *sbs_dev.cfg* for embedded systems that do not have a file system. Modify *dev_cfg.h* as follows:

1. Change directories to the *\sbs_ver<x.yz>\working* directory, where *<x.yz>* is the version number of the current release, if you are not already there.

2. Initialize the **`dev_cfg`** string array with the configuration parameters for each card. The keywords in *dev_cfg.h* are identical to those in *sbs_dev.cfg*; see Table 6.3.1 on page 40 for the keyword descriptions.

3. Save the file.

4. Refer to the Sample 1553 Applications Chapter of the *ABI/ASF User's Manual* for information on compiling your application.

An example of the parameters used for the *dev_cfg.h* file appears below for a dual-device ABI-V6 card running under vxWorks.

This example shows only the required parameters. In the actual *dev_cfg.h* file you would remove the comment (semicolon preceding the parameter and following the leading quotation mark). All remaining lines are left commented out (preceded by a semicolon after the leading quotation mark). If you are using a single-device card, you should uncomment keywords only for device 1. If you are running your card under an operating system other than the one in the example, refer to Table 6.3.1 to determine the keywords that your operating system requires.

Cross Reference: For an example of the complete *sbs_cfg.h* file, refer to the *Integrated Avionics Library Reference Manual*.

Note: The values required for your system, including the firmware filename, may differ from those shown below.

```
char *dev_cfg_array[] = {
"[DEVICE=1]",
"base_address=48000000h",
"device_type=M1553_V6_1",
"firmware=f005s.dat",
"int_vector=F0h",
"irq_level=1",
"[DEVICE=2]",
"base_address=48040000h",
"device_type=M1553_V6_2",
"firmware=f005s.dat",
"int_vector=F1h",
"irq_level=2"
};
```

6.4 Unit Test Using the Console Mode

This sections gives some of the basic procedures for using the Console Mode Unit Test. The procedures are as follows:

➢ Starting Unit Test Using the Console Mode
➢ Opening the 1553 Device
➢ Running Built-in Tests (BITs)
➢ Initializing the 1553 Device
➢ Exiting Unit Test

6.4.1 Starting Unit Test Using the Console Mode

Tips:
Press the carriage return key (**Enter**) to complete menu selections and enter responses to prompts.

Press **Enter** to restore a Unit Test menu following a failure.

1. Execute the Unit Test application.

   As soon as you execute Unit Test, it parses the configuration file (either *sbs_dev.cfg* or *dev_cfg.h*). If your configuration file parses without error, the menu shown in Figure 6.4.1 appears.

```
   SBS Technologies, Inc.
M1553 Interface Library Unit Test
  Version X.YZ    Build MMM DD YYYY

    1 - Device Management Tools
    2 - Bus Controller Tools
    3 - Remote Terminal Tools
    4 - Bus Monitor Tools

    q - Quit Unit Test

Selection ? >
```

Figure 6.4.1: M1553 Interface Library Unit Test Menu

Note: Where shown, X.YZ is the version number of the current release and MMM DD YYYY is the date the current build was compiled.

Note: If you are using the precompiled unit test, select **m** at Integrated Avionics Library Unit Test screen and press return. The *M1553 Interface Library Unit Test Menu as shown in* Figure 6.4.1 appears.

If a parser error occurs, an error screen may appear instead of the Avionics Interface Library Unit Test Menu. Figure 6.4.2 shows a sample parser error screen. Table 6.5.1 describes the common parser error messages that you may encounter.

```
                    SBS Technologies, Inc.
             Integrated Avionics Library Unit Test
                 Version X.YZ Build XX.YY.ZZ

               Failure parsing configuration file!
               Device #1 missing "firmware=" keyword.

               'Q' to quit, return to reparse.

               Selection ? >
```

Figure 6.4.2: Sample Parser Error Screen

2. If a parser error occurs, proceed to the troubleshooting procedures in Section 6.5.1.

   If no parser error occurs, the menu shown in Figure 6.4.1 appears.

6.4.2 Opening the 1553 Device

1. To select **Device Management Tools**, enter **1**.

   The menu shown in Figure 6.4.3 appears.

```
             1553 Device Management Tools              |              pg1
                                                       |
   1 - Init Device <Steps 2-7>   a - Get Device Clock  |    n - Next Pg->
   2 -  Open Device              b - Set Device Clock  |
   3 -  Load Firmware                                  |    p - Pick dev #
   4 -  Start Application        d - Display Error Tbl |
   5 -  Init Chan or V7 Mem      e - Clear   Error Tbl |    r - r/w Ram
   6 -  Init Interrupt Q                               |
   7 -  Create BSM Buffers                             |    s - Start i/o
                                                       |    t - sTop  i/o
   9 - Set ASF Mode              i - Execute BIT       |
   0 - Get Device Info           j - Close Device      |    x - eXit <-Pg
                                                       |
-------------------------------------------------------------------------------
Selection? >



Messages: Device # defaulted to 1.
-------------------------------------------------------------------------------

                    Device #1:  M1553_xxx_1  is  CLOSED                fyyy.dat
```

Figure 6.4.3: 1553 Device Management Tools Pg1 Menu

Note: Where shown, xxx is the card type (PCI, cPCI, etc.) of the card installed and fyyy.dat is the firmware being used.

Note: The device defaults to the first device number in the group of devices (in this case device #1). To select an alternate device number, enter **p**. At the “Device Number? >” message, enter the appropriate device number. The message “Device number changed to X. Hit ENTER to continue.” appears (X is the number of the selected device). Press Enter. The device number selected appears at the bottom of the 1553 Device Management Tools menu.

2. To select **Open Device**, enter **2**.

The messages shown in Figure 6.4.4 appear.

```
--------------------------------------------------------------------------------
Selection: 2

          Device #1 opened.
Messages: Hit ENTER to continue.
--------------------------------------------------------------------------------
```

Figure 6.4.4: Open Device Messages

3. To continue, press **Enter**.

The area between the dashed lines clears. The bottom line displays “Device #X: M1553_xxx_1 is OPEN” where X is the number of the device and xxx is the card type.

LEDs

- ➢ The RUN LED should be red (only valid immediately following powerup).
- ➢ The CH LED should be off.

Cross Reference: See Figure 4.2.1 on page 17 for locations and descriptions of the LEDs.

6.4.3 Running Built-in Tests (BITs)

1. To select **Execute Built-In Tests**, enter **`i`**.

   If you are prompted to enter a firmware source, enter the appropriate number to load from flash, from file, or via driver (depending on the type of card and operating system you are using, you may see only one of these options at the prompt).

   The messages and prompt shown in Figure 6.4.5 appear.

```
--------------------------------------------------------------------------------
Selection: i
For the built in test to work properly, a cable assembly with terminators
must be attached to the ABI/ASF card and the bus must be quiet.

Enter q to quit, or ENTER to start test.

Messages:
--------------------------------------------------------------------------------
```

Figure 6.4.5: Execute Built-in Tests Verification Screen

2. Verify that the cable assembly with terminators is attached to the ABI/ASF card (see page 19 for instructions).

3. Press **Enter**.

   The messages and prompt shown in Figure 6.4.6 appear.

```
--------------------------------------------------------------------------------
Selection: i
'0' = [FILE]; '1' = FLASH; '2' = DRIVER
Firmware source? >



Messages:
--------------------------------------------------------------------------------
```

Figure 6.4.6: Firmware Selection Screen

4. Enter the appropriate number to load from file, from flash, or via driver (depending on the type of card and operating system you are using, you may see only one of these options at the prompt) and press **Enter**.

   If the tests are successful, messages and prompts similar to Figure 6.4.7 will appear over a 10- to 12-second period.

```
--------------------------------------------------------------------------------
Selection: i
'0' = [FILE]; '1' = FLASH; '2' = DRIVER
Firmware source? > 1
Please Wait (Takes 6 to 9 seconds)...
           BIT passed!
           Device closed. Initialize before running.
Messages:  Hit ENTER to continue.
--------------------------------------------------------------------------------
```

Figure 6.4.7: Execute Built-in Tests Passed Screen

   If the tests are unsuccessful, error messages appear.

5. To continue (in either case), press **Enter**.

   The area between the dashed lines clears. The bottom line displays "Device #X: M1553_xxx_1 is CLOSED" where X is the number of the device and xxx is the card type.

LEDs

- ➢ The RUN LED should turn green.
- ➢ The CH LED should turn green briefly during the test, then turn off again.

Tips:
If the tests are unsuccessful, check for proper termination of the bus.

Contact SBS technical support if you are unable to correct the problem.

Cross Reference: See Figure 4.2.1 on page 17 for the locations and descriptions of the LEDs.

6.4.4 Initializing the 1553 Device

1. To select **Initialize Device**, enter **1**.

   The messages and prompts shown in Figure 6.4.8 appear one at a time.

```
-------------------------------------------------------------------
Selection: 1

Interrupt queue length [4]? >
Seq. monitor length [1000]? >
'0' = [FILE]; '1' = FLASH; '2' = DRIVER
Firmware Source? >

Messages:
-------------------------------------------------------------------
```

Figure 6.4.8: Initialize 1553 Device Prompts

2. To set the queue length to 4 entries, press **Enter** at the Queue Length prompt.
3. To set the sequential monitor length to 1000 words, press **Enter** at the Sequential Monitor Length prompt.
4. If you are prompted to enter a firmware source, enter the appropriate number to load from file, from flash, or via driver (depending on the type of card and operating system you are using, you may see only one of these options at the prompt).

   If the initialization process completes successfully, the message "Device initialized" appears. If this process does not complete successfully, an error appears between the dashed lines.
5. Press **Enter** to continue.

   The area between the dashed lines clears. The bottom line displays "Device #X: M1553_xxx_1 is STOPPED" where X is the number of the device and xxx is the card type.

LEDs

- The RUN LED should turn off briefly, then green again. (A "click" may be heard when the application firmware is loaded.)
- The CH LED should remain off.

Cross Reference: See Figure 4.2.1 on page 17 for locations and descriptions of the LEDs.

Troubleshooting If a failure occurs during initialization, proceed to Section 6.5.2.

6.4.5 Exiting Unit Test

1. To return to the M1553 Interface Library Unit Test menu, enter **x**.

   The screen clears, and the M1553 Interface Library Unit Test Menu appears.

2. To exit from the Unit Test application, enter **q**.

   The prompt shown in Figure 6.4.9 appears.

```
Are you sure you want to quit? ([y]/n) >
```

Figure 6.4.9: Prompt to Quit

Note: If you are using the precompiled unit test, the Integrated Avionics Library Unit Test screen appears next. Select **q** at Integrated Avionics Library Unit Test screen and press return. The prompt shown in Figure 6.4.9 appears.

3. Enter **y**.

6.5 Troubleshooting

This section discusses troubleshooting if a parser error occurs during the Unit Test using the Console Mode or if a failure occurs during initialization. The procedures are as follows:

- Parser Error
- Initialization Error

6.5.1 Parser Error

Unit Test Using Console Mode

1. Identify parser error using Table 6.5.1.
2. Correct the error in the configuration file. If you are using *sbs_dev.cfg*, press **Return** from the Unit Test error screen to reparse. If you are using *dev_cfg.h*, exit and recompile your code. Repeat step 1 in Section 6.4.

Note: After the configuration file parses without error (i.e., once you see the menu shown in Figure 6.4.1), you are ready to continue.

Table 6.5.1: Parser Error Messages

Error Message	Diagnosis
`Failure parsing configuration file!` `Error opening <filename> file!`	The indicated file could not be opened.
`Failure parsing configuration file!` `<keyword>=<num> for device #xx is invalid.`	The parser found a value out of limits in the configuration file. Correct the line containing `<keyword>`.
`Failure parsing configuration file!` `Application does not support <name> devices!`	The application type `<name>_APP` (where `<name>` is the avionics bus type) is not defined in the *sbs_sys.h* file.
`Failure parsing configuration file!` `Device #xx missing “<keyword>=” keyword.`	The specified `<keyword>`, which is required for the specified device (`Device #xx`, where `xx` is in the range 1 to `SBS_MAX_DEV`), was not found.
`Failure parsing configuration file!` `No device defined in configuration file!`	The `Device` keyword was not found in the configuration file.
`Failure parsing configuration file!` `“<name>” is not a valid device type.`	The `<name>` specified for the `device_type` keyword is not valid. Choose a valid device type from the list included in the configuration file.

`Failure parsing configuration file!`
`“<filename>” not defined in the`
`firmware.h file.`
	The `<filename>` specified for the `firmware` keyword in *dev_cfg.h* does not match any firmware filename in the *firmware.h* file.

`Failure parsing configuration file!` `“<keyword>=<num>” on line xx: duplicate entry.`	A value for the `<keyword>` specified on line `xx` has already been parsed for this device number.
`Failure parsing configuration file!` `Line number xx has more than 132 characters.`	The number of characters on line `xx` exceeds the maximum number that can be read.
`Failure parsing configuration file!` `“Device=0” on line xx must be greater than 0.`	The `Device` keyword must have a value greater than zero.
`Failure parsing configuration file!` `“Device=<num>” on line xx exceeds SBS_MAX_DEV.`	The value `<num>` is greater than the `SBS_MAX_DEV` value specified in the *dev_mgmt.h* file.

`Failure parsing configuration file!`
`“Device=<num>” is a noncontiguous <name>`
`2nd device.`
	The values of the `Device` keywords for the first and second devices of the indicated board (`<name>` = `PC3` or `PC16`) must be consecutive numbers.

6.5.2 Initialization Error

If a failure occurs during initialization, do the following:

1. Turn off the system, and then physically remove and reseat the card in the host computer.
2. Check all cable connections and verify that they are secure.
3. Try initializing the card again.
4. If a failure still occurs, refer to Table 6.5.2.

Table 6.5.2: Initialization Errors and Error Messages

Error	Diagnosis
Computer system locked up	Interrupt request (IRQ) level is in use. Specify a different IRQ level in the *sbs_dev.cfg* or *dev_cfg.h* file.
One of the following error messages appeared:	
`Initialize device failed!` `sbs_open_device(): The device driver failed to initialize`	The device driver was not started on reboot. Verify that the device is started in Windows NT or that the device driver is selected and loaded in Windows 95/98.
`Initialize device failed!` `sbs_open_device(): Shared memory fail`	The **base_address** specified in the *sbs_dev.cfg* or *dev_cfg.h* file is invalid or there is a problem with the operating system device driver. On UNIX systems, verify that the correct filename is specified for the **dd_name** keyword in *sbs_dev.cfg* or *dev_cfg.h*.
`Initialize device failed!` `sbs_load_ram(): Download error`	The **base_address** specified in the *sbs_dev.cfg* or *dev_cfg.h* file is invalid. For MS-DOS, Windows 3.x, and Windows 95/98/NT, verify that the required memory region is excluded in the *config.sys* file.
`Initialize device failed!` `sbs_load_ram(): File open error`	Your application could not open the firmware file specified in *sbs_dev.cfg* or *dev_cfg.h*. Verify that the firmware filename specified in *sbs_dev.cfg* or *dev_cfg.h* is correct and that the specified firmware file is in your *working* directory.
`Initialize device failed!` `sbs_start_firmware(): Start firmware failure`	The firmware did not properly start. Verify that the proper firmware file(s) are specified in *sbs_dev.cfg* or *dev_cfg.h*.
`Open device failed!` `sbs_open_device(): Device probe fail`	It is possible that the PCI device driver may be conflicting with the resources of another device. Try changing the memory setting for the PCI device in system resources. Conflicts with video display adapters using ROM Shadowing have resulted in resource conflicts.

Cross Reference: For further help with initialization failures, see *Customer Support Services* in the Introduction Chapter of the *MIL-STD-1553ABI/ASF User's Manual*.

7: Connecting to the 1553 Bus

If the Unit Test procedures above complete without error, you may proceed with MIL-STD-1553 operations. However, you must first connect each channel of the card to an actual 1553 bus by completing the following procedure:

1. Remove the terminators from the Bus A and Bus B cables (these cables are marked “Channel 1A” and “Channel 1B,” respectively).

2. Attach the Bus A cable to the stub connector on an appropriate transformer-coupled device (see Figure 7.0.1).

3. Attach the Bus B cable to the stub connector on an appropriate transformer-coupled device (see Figure 7.0.1).

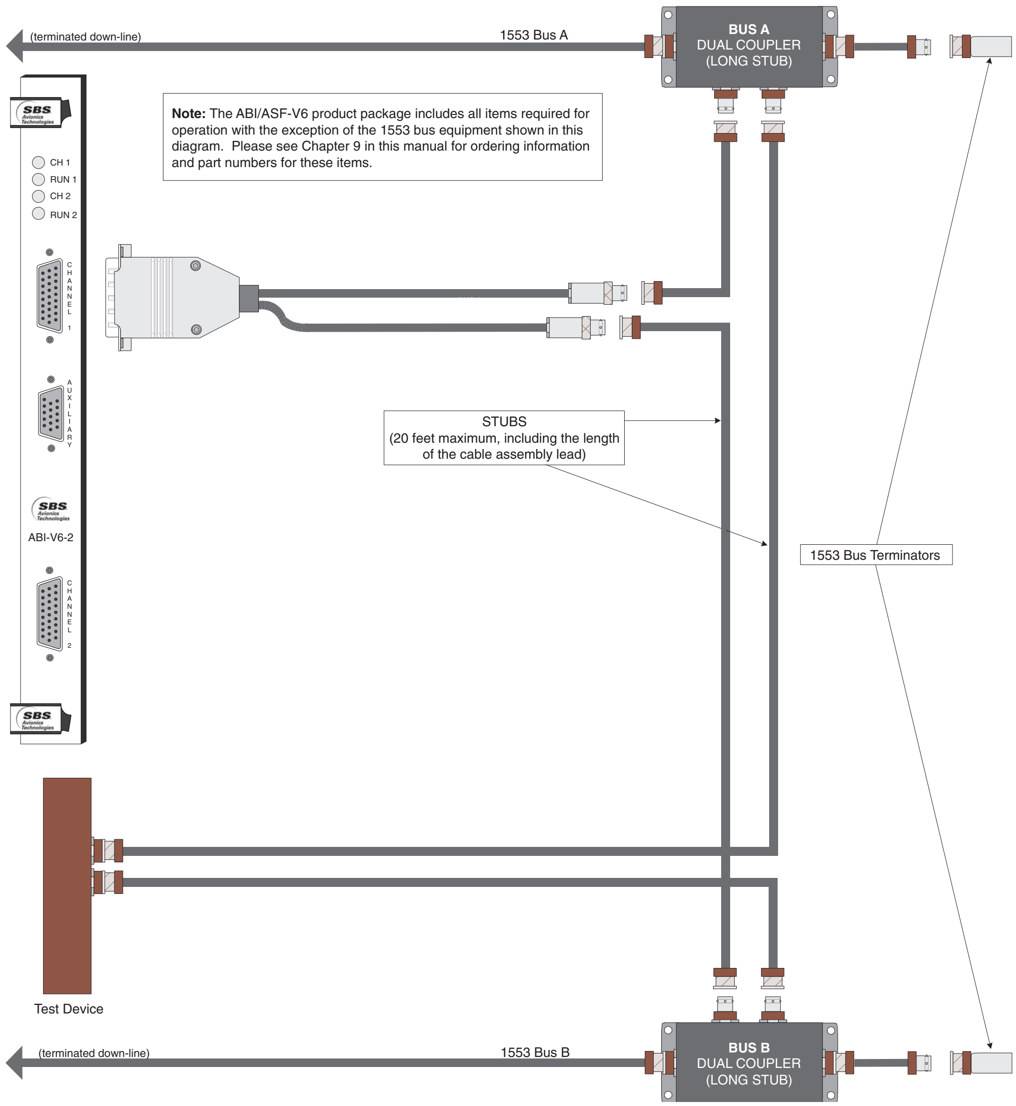


Figure 7.0.1: Connecting the V6 to a 1553 Bus

8: What to Do Next

If your card is properly connected to a 1553 bus, you may proceed in one or more of the following ways:

➢ Run sample 1553 applications.

Cross Reference: If you are a first-time user, SBS also recommends that you execute the sample 1553 applications that are included with the product shipment. These applications are detailed in Sample 1553 Applications Chapter 0f the *MIL-STD-1553 ABI/ASF User's Manual*.

➢ Build a custom 1553 application.

Cross Reference: If you are an advanced user, you may wish to begin designing your own 1553 application. See Chapters 4–15 in the *MIL-STD-1553 ABI/ASF User's Manual* for complete details.

9: Hardware Specifications

This chapter presents information about the ABI/ASF-V6-1 and ABI/ASF-V6-2 hardware, including:

- General Product Information
- Physical Specifications
- Operational Specifications

The General Product Information section contains general information for the ABI/ASF-V6-1 and ABI/ASF-V6-2 cards, including the cage code number, extended warranty information, conformance to MIL-STD-1553 electrical specifications, bus equipment part numbers and ordering instructions, and external signal characteristics.

In the Physical Specifications section, you can find information about component locations, dimensions, jumper and DIP switch settings, connectors, pinouts diagrams, and where you can order connecting cables and terminators.

The Operational Specifications section contains information about the operational aspects of the card, including temperatures, MTBF, and hardware reset.

9.1 General Product Information

The information in this section applies to the ABI/ASF-V6-1 and ABI/ASF-V6-2 hardware. This section contains the following topics:

- Cage Code Number
- Extended Warranty Information
- Conformance to MIL-STD-1553 Electrical Specifications
- Bus Equipment Part Numbers and Ordering Instructions
- External Signal Characteristics

9.1.1 Cage Code Number

0BAS8

9.1.2 Extended Warranty Information

SBS offers a comprehensive maintenance service for the ABI/ASF products. Even though SBS boards rarely fail, these services assure that the end user has thorough coverage and minimal down time in case of a failure.

SBS products include a two-year, parts and labor warranty. You can purchase an extended warranty to extend this service beyond the second year. This provides you with 10-day turn-around for the repair of a module (or provides a replacement module at no cost). Large-quantity repairs may require a longer turn-around time. The cost is minimal and agreements are normally for one-year periods.

For more information or to receive a copy of the maintenance agreement, contact SBS Technologies at one of the numbers listed on the inside cover of this manual and specify “Warranty Support.”

9.1.3 Conformance to MIL-STD-1553 Electrical Specifications

All SBS products conform to the 1553 electrical specifications illustrated in Figure 9.1.1.

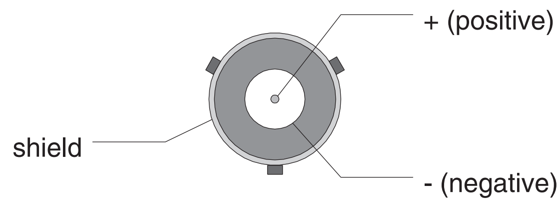


Figure 9.1.1: 1553 Electrical Specifications (as Viewed from End of Connector)

9.1.4 Bus Equipment Part Numbers and Ordering Instructions

You can purchase the bus equipment listed in Table 9.1.1 from SBS Technologies.

Table 9.1.1: OPE 1553 Bus Hardware Available from SBS Technologies

Part Number	Description
BUS-2	Dual-redundant bus with dual-stub couplers: four 15’-cables, four terminators, two 2-stub couplers
BUS-3	Dual-redundant bus with three-stub couplers: six 15’-cables, four terminators, two 3-stub couplers
Single Stub	Single-stub bus coupler
2 Stub	2-stub bus coupler
3 Stub	3-stub bus coupler
4 Stub	4-stub bus coupler
TERM	78-ohm terminator
CAB-COM-ZZ	Commercial-grade cable with PL-75 connectors (ZZ=length in feet)
CAB-MIL-ZZ	MIL17-grade cable with PL-75 connectors (ZZ=length in feet)
BUS-R	Regenerates bus signals for extending a MIL-STD-1553B Notice 2 compliant bus by 100 meters or 330 feet. Two BUS-R products are required for a dual-redundant bus.
BUS-C	Provides an RS-422 interface to SBS ABI modules to extend a MIL-STD-1553B stub from 20 to 300 feet
RS422-C	Cable Option: 300-ft RS-422 Cable for BUS-C

For more information, contact SBS at one of the numbers listed on the inside cover of this manual and specify “Sales Support.”

Miscellaneous 1553 components are also available from the following vendor:

MilesTek
1506 Interstate 35 W
Denton, Texas 76207-2402
Attn: Al Stenzel
800-524-7444 or 940-484-9400
FAX: 940-484-9402

9.1.5 External Signal Characteristics

External Trigger The external trigger feature is standard on the ABI/ASF-V6-1 and ABI/ASF-V6-2. The external trigger is a transistor-transistor logic (TTL) signal having the characteristics shown in Figure 9.1.2.

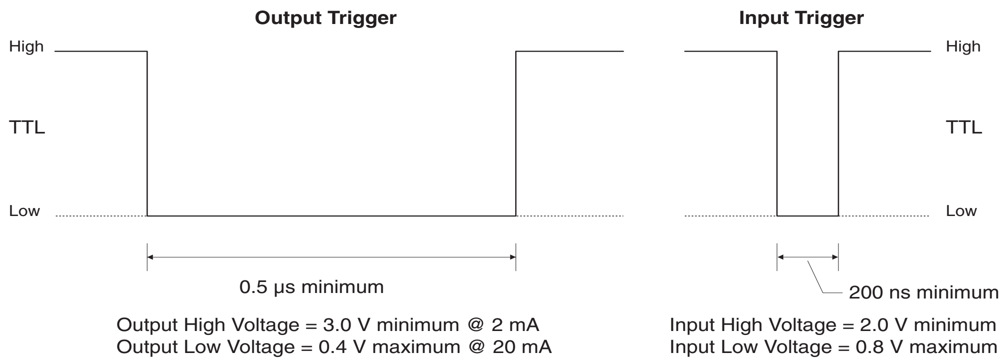


Figure 9.1.2: Characteristics of ABI/ASF External Trigger

Note: When the low or high level is selected for the input trigger, the signal must remain at that level for at least 20 microseconds. Otherwise, the firmware may not delete the input trigger.

External Clock The ABI/ASF-V6-1 and ABI/ASF-V6-2 firmware supports the external clock feature. The external clock requires a differential signal with the specifications listed in Table 9.1.2.

Table 9.1.2: External Clock Differential Signal Specifications

Specification	Value
Recommended Differential Input Voltage (measured from + External Clock pin to – External Clock pin)	+12 V
Recommended Input Voltage (measured at either + pin or – pin)	Minimum: –7 V Maximum: +12 V
Differential Input Threshold	Minimum: –0.2 V Maximum: +0.2 V
Typical Output Voltages	High (minimum): +2.7 V Low (maximum): +0.5 V

IRIG The IRIG clock feature is an option which must be ordered at the time of purchase. It is available on all ABI/ASF products. ABI/ASF products which include this option will accept an IRIG input signal compatible with the IRIG-B standard.

Note: The IRIG input impedance for all ABI/ASF products is 10 kΩ.

9.2 Physical Specifications

This section describes the physical specifications for the ABI/ASF-V6-1 and ABI/ASF-V6-2 cards. This section contains the following topics:

- ➢ Board Layout
- ➢ Board Dimensions
- ➢ Switch and Jumper Settings
- ➢ Examples of Various Base Memory Address/Address Mode Combinations
- ➢ Setting the Switches
- ➢ Connector Descriptions and Pinouts
- ➢ Front Panel Pinouts and LEDs
- ➢ Part Numbers and Ordering Instructions

9.2.1 Board Layout

Figure 9.2.1 shows the layout of the ABI/ASF-V6-2 board.

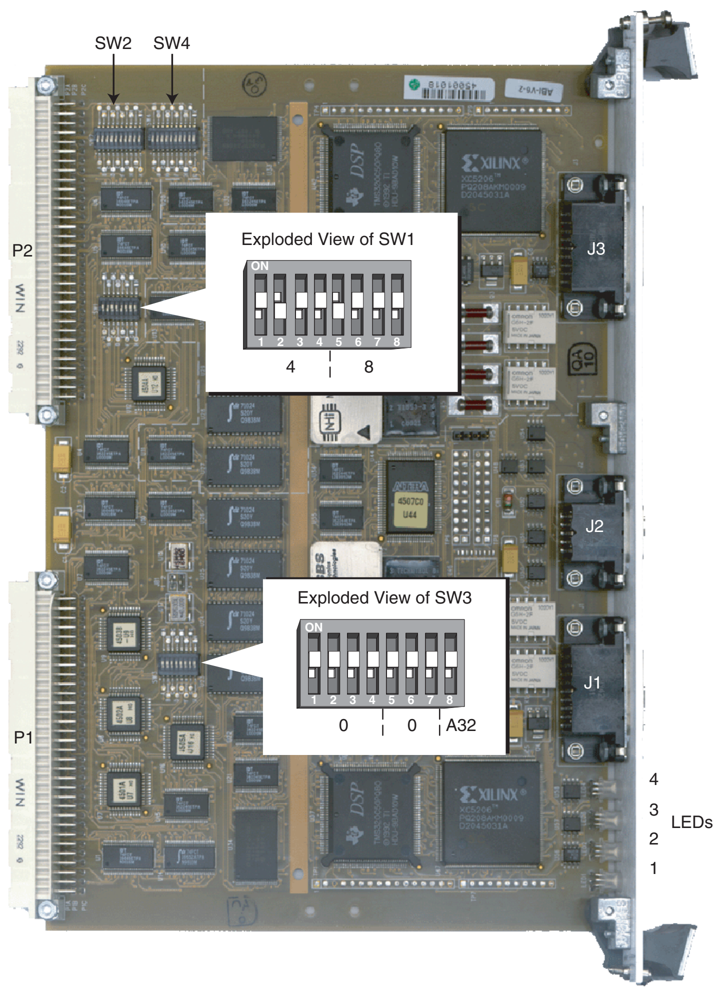


Figure 9.2.1: ABI/ASF-V6-2 Board Layout

9.2.2 Board Dimensions

Standard 6u x 160mm size

9.2.3 Switch and Jumper Settings

Base Address/ Address Mode

Table 9.2.1 describes the V6-1 and V6-2 jumper and switches. Table 9.2.2 lists the base address and address mode settings for base address 4800 0000h and address mode A32.

Table 9.2.1: V6 Jumper Block and Switches

Jumper Block or Switches	Description	Default Setting
JB1	Configures clock type (commercial or industrial)	Hardware Dependent **Warning:** DO NOT ALTER!
SW1, SW3	Base Address/Address Mode Configuration	4800 0000h/A32
SW2, SW4	Optional P2 Connections for 1553 and external clock. See Table 9.2.4.	None

Table 9.2.2: V6 Base Address/Address Mode Selection (Illustrating Base Address 4800 0000h and Address Mode A32)

Address Bit	Example Value	Switch Location	Switch Setting
A31	0	SW1-1	ON
A30	1	SW1-2	OFF
A29	0	SW1-3	ON
A28	0	SW1-4	ON
A27	1	SW1-5	OFF
A26	0	SW1-6	ON
A25	0	SW1-7	ON
A24	0	SW1-8	ON
A23	0	SW3-1	ON
A22	0	SW3-2	ON
A21	0	SW3-3	ON
A20	0	SW3-4	ON
A19	0	SW3-5	ON
A18	0	SW3-6	ON
A17	0	SW3-7	ON
A32/A24 Select[a]	0	SW3-8	ON

[a] A32 address mode is selected by setting SW3-8 ON; A24 address mode is selected by setting SW3-8 OFF. When you are using A24 addressing, the positions of the switches on SW1 are not applicable.

9.2.4 Examples of Various Base Memory Address/Address Mode Combinations

Figure 9.2.2 and Figure 9.2.3 illustrate two base addresses with address mode A32. Figure 9.2.4 and Figure 9.2.5 illustrate two base addresses with address mode A24.

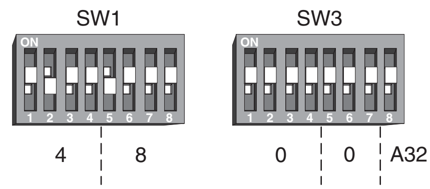


Figure 9.2.2: Base Memory Address 4800 0000h, Address Mode A32

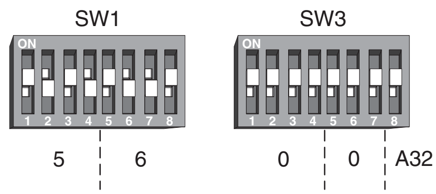


Figure 9.2.3: Base Memory Address 5600 0000h, Address Mode A32

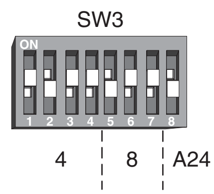


Figure 9.2.4: Base Memory Address 48 0000h, Address Mode A24

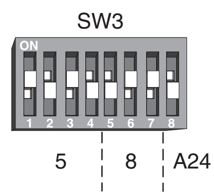


Figure 9.2.5: Base Memory Address 58 0000h, Address Mode A24

9.2.5 Setting the Switches

Figure 9.2.1 shows the layout of the V6-2 board. The V6-1 board is similar but lacks the J3 connector and LEDs 3 and 4. Using Figure 9.2.1, locate switches SW1 and SW3 on the V6 module. Next set the switch positions to select the desired base memory address and address mode. See the specific directions below for setting the switch positions for A32 and A24 address modes.

Note: Toggle a switch position to "on" to set the value of the corresponding address bit to "0"; toggle a switch position to "off" to set the value of the corresponding address bit to "1".

A32 Address Mode

Two switches, SW1 and SW3, define the base memory address and address mode settings for the A32 address mode on the V6. If you are using the A32 address mode, perform the following steps to properly set these switches:

1. Set SW3 position 8 to "on".
2. Set SW1 positions 1–8 and SW3 positions 1–7 to the base memory address you identified above.

   Figure 9.2.2 and Figure 9.2.3 illustrate the switch settings for base memory addresses of 4800 0000h and 5600 0000h, respectively.

Note: The values represented below are for illustration only and do not necessarily represent a valid base memory address or address mode for your system.

Base Address Boundary Considerations

The Base Address for the card needs to be on 512-k boundaries. For example, two consecutive base addresses could be XXX00000h and XXX80000h. The memory address range for channel 1 is from XXX00000h to XXX3FFFFh, while the addressing for channel 2 is from XXX40000h to XXX7FFFFh (i.e. A18 = "0" for channel 1 and A18 = "1" for channel 2). Notice we do not use address A17 (SW3-7) in setting the base address.

A24 Address Mode

One switch, SW3, defines the base memory address and address mode settings for the A24 address mode on the V6. If you are using the A24 address mode, perform the following steps to properly set this switch:

1. Set SW3 position 8 to "off".
2. Set SW3 positions 1–7 to the base memory address you identified above.

Figure 9.2.4 and Figure 9.2.5 illustrate the switch settings for base memory addresses of 48 0000h and 58 0000h, respectively.

9.2.6 Connector Descriptions and Pinouts

Table 9.2.3 lists the connectors and switches on the V6-1 and V6-2. Table 9.2.4 lists the 1553 signals and external timing controls, which are hard wired to the front panel connectors. All of the front panel connector pins not listed in the table are reserved. These signals may also be connected to the user I/O pins on the VME P2 connector via switches SW2 and SW4. The board ships with these signals disconnected. Table 9.2.4 provides the signal description with the switch number and corresponding VME P2 pin designation.

Table 9.2.3: V6 Connectors and Related Switches

Connector	Description
J1	1553 Channel 1 Bus (Primary and Secondary)
J2	Assorted signals for external input/output connections
J3	1553 Channel 2 Bus (Primary and Secondary) **Note:** This connector is available only on dual-channel cards.
P1, P2	VMEbus Interface
SW2, SW4	Enables optional P2 connections for 1553 and external clock

Table 9.2.4: V6 Pinouts for J1, J2, J3, and P2 Connectors

Front Panel Connector Pin	Description	Switch No.	Default Switch Setting	P2 pin #
J2-9	GND	N/A	N/A	N/A
J2-11	GND	N/A	N/A	N/A
J2-13	External Clock In +	SW2-1	OFF	C8
J2-14	External Clock In –	SW2-2	OFF	C9
J2-3	Reserved	SW2-3	OFF	C10
J2-4	Reserved	SW2-4	OFF	C11
J2-6	Reserved	SW2-5	OFF	C14
J2-7	Reserved	SW2-6	OFF	C15
J2-8	External Trigger	SW2-7	OFF	C16
J2-10	Reserved	SW2-8	OFF	C17
J2-12	Reserved	SW2-9	OFF	C20
N/A	AUTOBOOT (See Warning and Note below table.)	SW2-10	OFF	N/A
J1-1	Channel 1 Bus / Primary +	SW4-1	OFF	A8
J1-2	Channel 1 Bus / Primary –	SW4-2	OFF	A9
J1-3	Channel 1 Bus / Secondary +	SW4-3	OFF	A10
J1-4	Channel 1 Bus / Secondary –	SW4-4	OFF	A11
J3-1	Channel 2 Bus / Primary +	SW4-5	OFF	A14

Front Panel Connector Pin	Description	Switch No.	Default Switch Setting	P2 pin #
J3-2	Channel 2 Bus / Primary –	SW4-6	OFF	A15
J3-3	Channel 2 Bus / Secondary +	SW4-7	OFF	A16
J3-4	Channel 2 Bus / Secondary –	SW4-8	OFF	A17
J2-1	P2IRIG	SW4-9*	OFF	C4
J2-2	P2IGND	SW4-10*	OFF	C5

* Switches SW4-9 and SW4-10 were added in board revision D.

Warning: The autoboot switch is used for special purposes and must normally be switched off.

Note: To enable the Autoboot feature, set SW2 position 10, to the "on" position.

Table 9.2.5 and Table 9.2.6 provide the VME P1 and P2 connections for the V6. Table 9.2.6 also provides the available SW2 and SW4 connections, shaded in gray.

Table 9.2.5: VMEbus P1 Connections

Pin	Row A	Row B	Row C
1	D00	BBSY*	D08
2	D01	BCLR*	D09
3	D02	N/C	D10
4	D03	BGIN0*	D11
5	D04	BGOUT0*	D12
6	D05	BGIN1*	D13
7	D06	BGOUT1*	D14
8	D07	BGIN2*	D15
9	GND	BGOUT2*	GND
10	N/C	BGIN3*	N/C
11	GND	BGOUT3*	BERR*
12	DS1*	BR0*	RESET*
13	DS0*	BR1*	LWORD*
14	WR*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	N/C	A17
22	IACKOUT*	N/C	A16

* Active low signal

Pin	Row A	Row B	Row C
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	N/C	N/C	+12 VDC
32	+5 VDC	+5 VDC	+5 VDC

* Active low signal

Table 9.2.6: VMEbus P2 Connections (SW2 and SW4 connections Shaded)

Pin	Row A	Row B	Row C
1	N/C	+5 VDC	N/C
2	N/C	GND	N/C
3	N/C	N/C	N/C
4	N/C	A24	N/C
5	N/C	A25	N/C
6	N/C	A26	N/C
7	N/C	A27	N/C
8	CH 1 BUS PRI +	A28	EXT CLK +
9	CH 1 BUS PRI –	A29	EXT CLK –
10	CH 1 BUS SEC +	A30	EXT TRG 1+
11	CH 1 BUS SEC –	A31	EXT TRG 1–
12	N/C	GND	N/C
13	N/C	+5 VDC	N/C
14	CH 2 BUS PRI +	D16	EXT TRG 2+
15	CH 2 BUS PRI –	D17	EXT TRG 2–
16	CH 2 BUS SEC +	D18	EXT TRG 3
17	CH 2 BUS SEC –	D19	EXT TRG 4
18	N/C	D20	N/C
19	N/C	D21	N/C
20	N/C	D22	EXT TRG 5
21	N/C	D23	N/C
22	N/C	GND	N/C
23	N/C	D24	N/C
24	N/C	D25	N/C
25	N/C	D26	N/C
26	N/C	D27	N/C
27	N/C	D28	N/C
28	N/C	D29	N/C

Pin	Row A	Row B	Row C
29	N/C	D30	N/C
30	N/C	D31	N/C
31	N/C	GND	N/C
32	N/C	+5 VDC	N/C

9.2.7 Front Panel Pinouts and LEDs

Figure 9.2.6 shows the front panel pinouts and describes the LEDs.

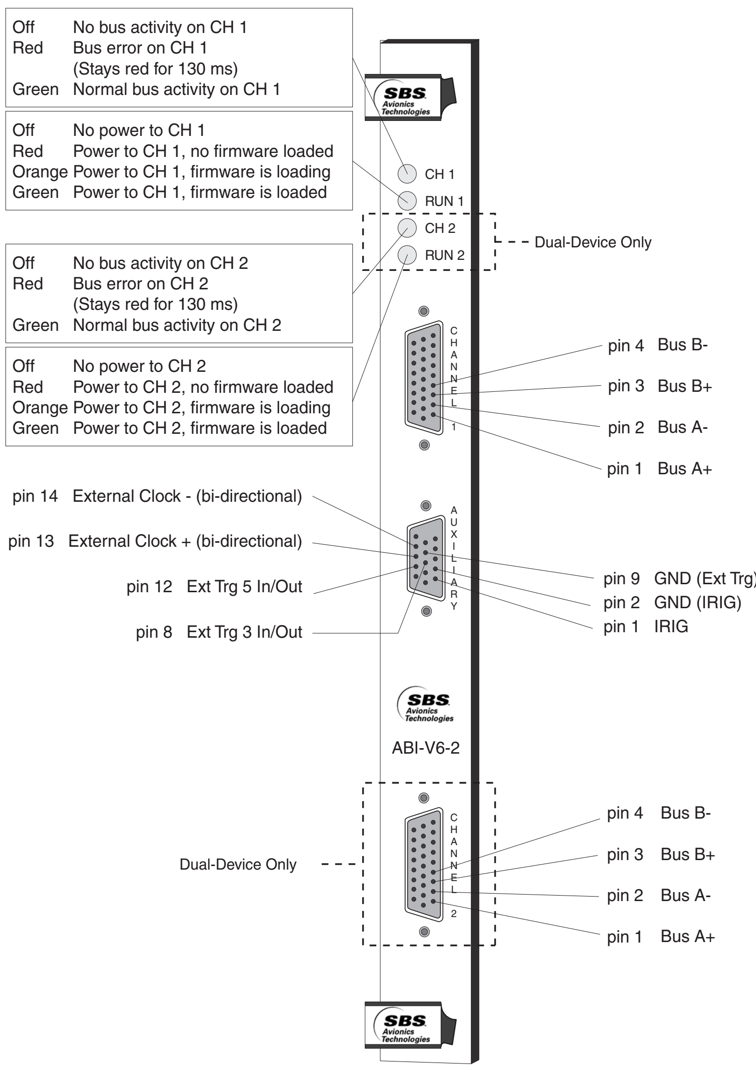
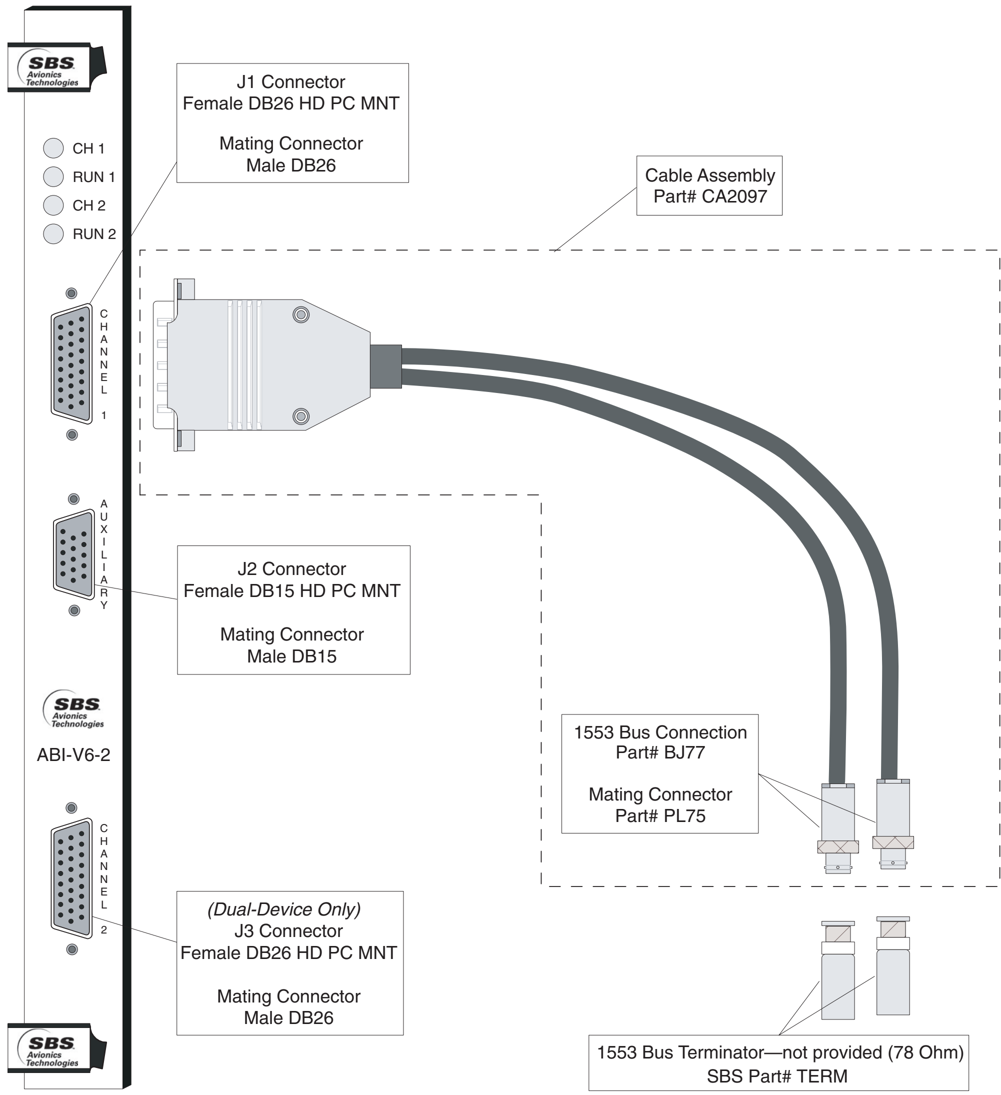


Figure 9.2.6: ABI/ASF-V6 Front Panel Pinouts and LEDs

9.2.8 Part Numbers and Ordering Instructions

SBS provides one cable assembly, part number CA2097, for each channel on the V6. This assembly attaches to the J1 (Channel 1) or J3 (Channel 2) connector on the V6 front panel. It provides cable leads to make connections to Bus A and Bus B.

Figure 9.2.7 illustrates the cable assembly and provides part numbers and ordering instructions for all external connectors on the V6.



Ordering Instructions

1553 equipment may be purchased from one of the following:

SBS Technologies, Inc.
800-SBS-1553 or 505-875-0600
FAX: 505-875-0400

Milestek
1506 Interstate 35 W
Denton, TX 76207-2402
Attn: Al Stenzel
800-524-7444 or 940-484-9400
FAX: 940-484-9402

Standard connectors may be purchased from the following vendor:

DigiKey
800-344-4539
FAX: 218-681-3380

Figure 9.2.7: Part Numbers and Ordering Instructions for V6 Connectors

9.3 Operational Specifications

The information in this section presents the operational specifications for the ABI/ASF-V6-1 and ABI/ASF-V6-2. This section contains the following topics:

- VMEbus Characteristics
- Temperature
- Shock
- MTBF
- Hardware Reset

9.3.1 VMEbus Characteristics

The V6 board supports slave data transfers in D64, D32, and D16 modes. It supports single transfers in all D32 and D16 modes, and supports block transfers in either D64 or D32 mode.

The V6 board supports both Supervisor and Nonprivileged address modes. The board also supports both A32 and A24 VMEbus address modes. You switch-configure the desired VMEbus address mode.

All Interrupt vectors are type D08.

9.3.2 Temperature

Table 9.3.1 lists the V6 operating and storage temperature specifications.

Table 9.3.1: V6-1 and V6-2 Temperature Specifications

Type of Card	Operating	Storage
Commercial	0° to +60° Celsius	–65° to +150° Celsius
Industrial	–40° to +85° Celsius	–65° to +150° Celsius

9.3.3 Shock

The following are available upon request:

- Full ruggedization with stiffener
- Optional conformal coating

9.3.4 MTBF

Table 9.3.2 lists the mean time between failures for the V6. We calculated the MTBF using the MIL-HDBK-217F, Parts Count Method - Ground Benign Environment.

Table 9.3.2: V6-1 and V6-2 Mean Time Between Failures

Type of Card	V6-1	V6-2
Commercial	122,826 hours	70,882 hours
Industrial	153,593 hours	88,260 hours

9.3.5 Hardware Reset

If the hardware reset button on the VMEbus system is pressed, the CSR resets and firmware execution halts. At this point, memory above 003Fh is still intact and may be accessed. After a hardware reset, you must restart the firmware and the memory above 003Fh clears at this time.

Cross Reference: See Section 10.2 for a description of the CSR.

10: Card Information

Note: The terms channel and device are used interchangeably in this manual.

This chapter provides card information for the ABI/ASF-V6-1 and ABI/ASF-V6-2 cards on memory organization, hardware control registers, and downloading/starting the firmware. This chapter covers the following topics:

- Memory Organization
- Hardware Control Registers
- Methods of Downloading the Firmware File
- Downloading the Firmware File
- Starting the Firmware
- Reprogramming the Flash Memory

10.1 Memory Organization

As shown in Figure 10.1.1, the V6 memory map divides into two independent 256-kilobyte sections, one for each channel. You access channel 1 registers and firmware structure pointers from word offset 00000h, and channel 2 registers and pointers from word offset 20000h.

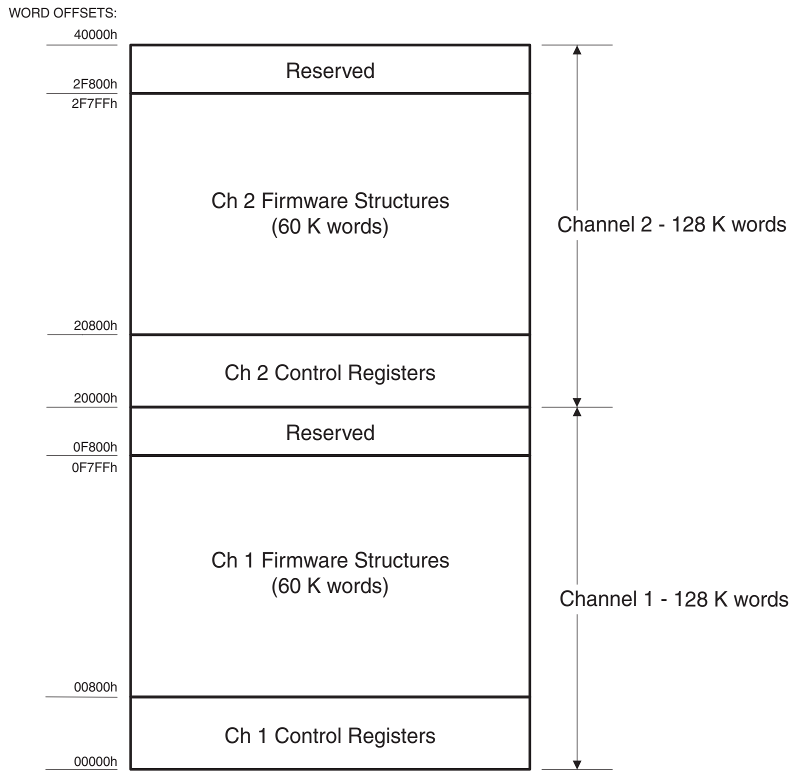


Figure 10.1.1: V6 Memory Map

10.2 Hardware Control Registers

This section presents bit settings for the hardware control registers for the V6 card as follows:

- CSR
- INTV_1553
- INTV_1553 Gen Inter

These registers provide the following module functions to the host:

- Module reset and operation control
- VMEbus interrupt control
- DSP interrupt control

Warning: Accessing addresses not described (03–3Fh) may adversely affect V6 operation and produce unpredictable results. Do not access these registers!

10.2.1 CSR

The Control/Status Register (word address 00h, byte address 000h) is a key register for proper initialization and operation of the V6. This is the only hardware control register always accessible to the host. Table 10.2.1 shows the bits for this register.

Table 10.2.1: V6 CSR Bit Descriptions

Bit	Function*	Description							
0	Run	0 = Firmware stop 1 = Firmware run							
1	Code Location Select	0 = Load from data RAM 1 = Load from FLASH memory							
2	Reserved	Reserved for factory test purposes. *Clear this bit when writing to the CSR.*							
3	VMEbus Interrupt Enable	0 = Clear 1 = Enable interrupt When this bit is enabled, an interrupt can be generated by the V6 with the corresponding vector in register word offset 01h.							
4	Interrupt Pending (ro) Interrupt Clear (wo)	0 = No interrupt 1 = Interrupt pending (read)/clear interrupt (write) Read this bit to determine whether a VMEbus interrupt is pending. Set this bit to clear the interrupt.							
		Interrupt Level (Set bits 5–7 for the desired interrupt level)							
		Level 0	**Level 1**	**Level 2**	**Level 3**	**Level 4**	**Level 5**	**Level 6**	**Level 7**
5	Interrupt Level Select 0	0	1	0	1	0	1	0	1
6	Interrupt Level Select 1	0	0	1	1	0	0	1	1
7	Interrupt Level Select 2	0	0	0	0	1	1	1	1
8–15	Reserved	–							

*Except where otherwise noted, the host has both read and write access to the CSR bits.
ro = Read only access
wo = Write only access

10.2.2 INTV_1553

Bits 0–7 of this register (word address 01h, byte address 002h) set the VMEbus Interrupt Vector for 1553 user-selected interrupts for the V6. Bit 0 is the least-significant bit.

Note: The host has both read and write access to the INTV_1553 bits.

10.2.3 INTV_1553 Gen Inter

Write any data to this register (word address 02h, byte address 004h) to cause a VMEbus interrupt (having the vector specified by word address 01h) to occur. Factory testing uses this, but you may also use this to verify proper VME ISR operation.

Note: The host has write only access to the INTV_1553 Gen Inter bits.

10.3 Methods of Downloading the Firmware File

The V6 requires that you download all firmware code prior to initializing application data structures. The three methods available for downloading the firmware code are:

- ➢ Host Download
- ➢ Flash Memory Download (Under control of the host)
- ➢ Autoboot (Automatic flash memory download when the board powers up)

10.3.1 Host Download

For the host download, a V6 download file contains the firmware code which comes in ASCII format. The data in the download file has a 16-bit word structure. The first sixteen words make up the file header, containing product and version information. The seventeenth word in the file contains a word count value (N) for the first half of the data in the file. See Figure 10.3.1.

10.3.2 Flash Memory Download

For the host-controlled flash memory download, load the firmware automatically through the use of two bits in the CSR register. See the subsection *Starting the Firmware from Flash Memory* on page 80 for more information.

10.3.3 Autoboot

For autoboot, configure the V6 to automatically load firmware from flash memory when the board powers up. The autoboot configuration is set via a switch for the V6.

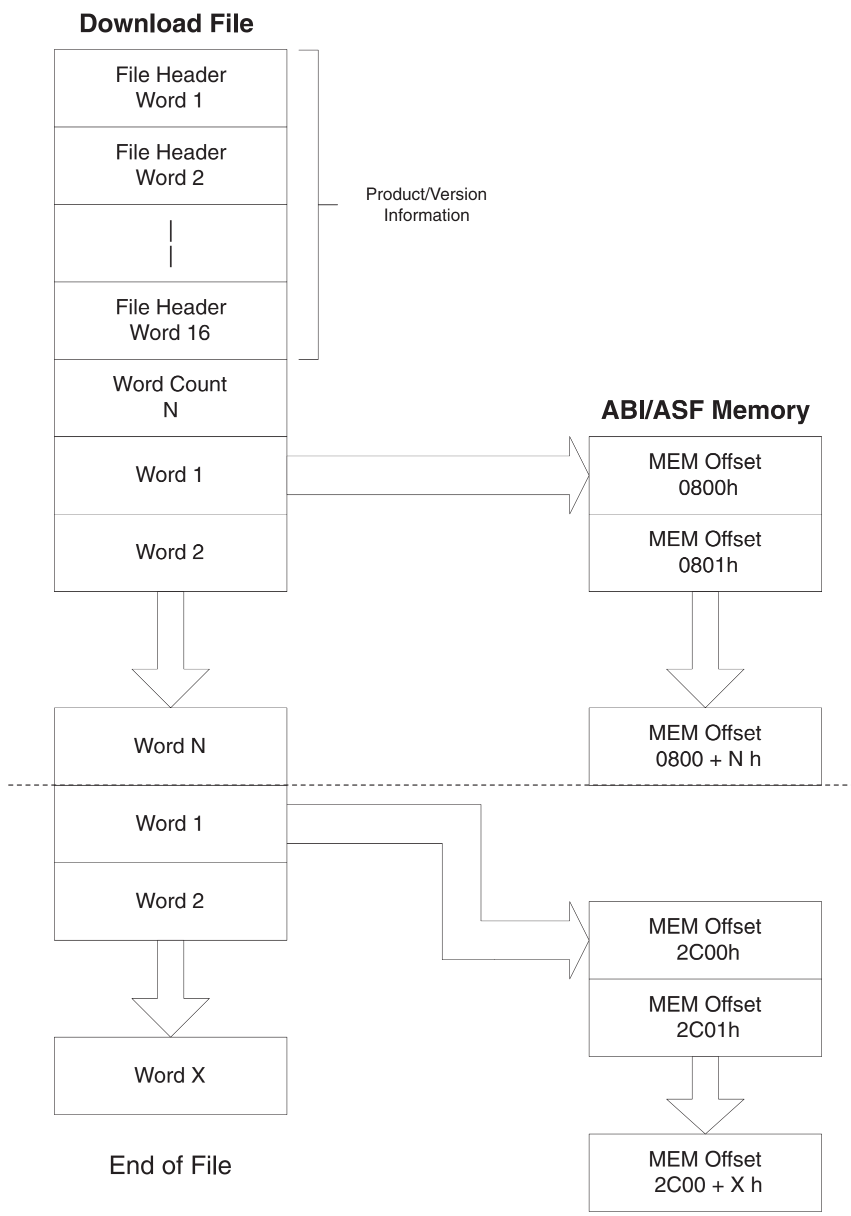


Figure 10.3.1: V6 Download File Format

10.4 Downloading the Firmware File

Complete the following instructions to download the firmware file to the V6:

1. Open the firmware code file (text mode).
2. Using the I/O Control/Status Register, verify that the internal processor is not running.
3. Read and skip the first sixteen words in the ASCII file.
4. Read the seventeenth word (N). This is the number of data words in the first half of the file.
5. Starting at offset 00800h, read the next word from the file and write the word to memory.
6. Continue reading the file, writing the data, and incrementing the address until N words have been read and written.
7. After the Nth word is processed, repeat steps 4 and 5 with a starting offset of 02C00h until the end of the file is reached.

10.5 Starting the Firmware

This section describes the methods of starting the firmware for the ABI/ASF-V6-1 and ABI/ASF-V6-2 cards. It covers the following topics:

- ➢ Starting the Firmware from On-Board RAM
- ➢ Starting the Firmware from Flash Memory
- ➢ Starting the Firmware after Autoboot

10.5.1 Starting the Firmware from On-Board RAM

After powering up or resetting the V6 module, first download the code/data file per the firmware download instructions. Upon completion of the download, perform the following procedure to start up the V6:

1. Write 0000h to the CSR (offset 0) to reset the V6 hardware.
2. Write 000Ah to offset 07FFh.
3. Write FFFFh to the BIT Status register (offset 3Bh).
4. Write 0001h to the CSR to start the V6 firmware.
5. Read the BIT Status register (offset 3Bh) and wait for the value to equal 0000h, indicating that the power-up tests have completed.
6. Read the BIT total error count (offset 3Ch). The value will be nonzero if errors were detected. If this occurs, contact SBS Technologies' Technical Support.

After you complete this procedure, the V6 is in the BIT mode awaiting a command. Either select BIT tests to perform BIT or initialize the board for 1553 operations.

10.5.2 Starting the Firmware from Flash Memory

After powering up or resetting the V6 module, complete the following steps to start up the V6 using the flash memory:

1. Write 0002h to the CSR (offset 0) to reset the V6 hardware.
2. Write FFFFh to the BIT Status register (offset 3Bh).
3. Write 0003h to the CSR to start the V6 firmware.
4. Read the BIT Status register (offset 3Bh) and wait for the value to equal 0000h, indicating that the power-up tests have completed.
5. Read the BIT total error count (offset 3Ch). The value will be nonzero if errors were detected. If this occurs, contact SBS Technologies.

After you complete this procedure, the V6 is in the BIT mode awaiting a command. Either select BIT tests to perform BIT or initialize the board for 1553 operations.

10.5.3 Starting the Firmware after Autoboot

After powering up the V6 module, the V6 automatically downloads firmware from flash memory and then runs BIT. Complete the following steps to start up the V6:

1. Read the BIT Status register (offset 3Bh) and wait for the value to equal 0000h, indicating that the power-up tests are complete.
2. Read the BIT total error count (offset 3Ch). The value is nonzero if errors were detected. If the value is nonzero, contact SBS technical support.

After you complete this procedure, the V6 is in the BIT mode awaiting a command. Either select BIT tests to perform BIT or initialize the board for 1553 operations.

Note: When configured to autoboot, the V6 cannot have the firmware downloaded via the host, nor can it be commanded to download firmware under the control of the host. When configured to autoboot, the V6 downloads firmware from flash memory only when the board powers up.

10.6 Reprogramming the Flash Memory

You can reprogram the V6 flash memory, providing memory updates at your site without the need for PROM replacements. The procedure is similar to the three methods of module startup previously described. Complete the following steps:

1. Perform a Software Download with the new code to be programmed.
2. Perform a Module Startup - Data RAM Mode. This will verify that the code is loaded into the V6 memory before reprogramming the Flash.
3. Perform the Software Download with the new code to be programmed.
4. Perform a Module Startup - Flash Mode using the following steps:
5. Write 0002h to the CSR (offset 0) to reset the V6 hardware.
6. Write C0DEh to Program Command Register 1 (offset 0400h).
7. Write 1234h to Program Command Register 2 (offset 0401h).
8. Write FFFFh to the BIT Status Register (offset 3Bh).
9. Write 0003h to the CSR to start the V6 firmware.
10. Read the BIT Status Register and wait for the value to equal 0000h, indicating that the power-up tests have completed. Completion of this step requires up to five seconds.
11. Read both Program Command Registers (offsets 0400h and 0401h). The values will be nonzero if flash programming errors were detected.

This procedure reprograms the flash memory with the new code, which is then used to restart the V6. After you complete this procedure, the V6 is in the BIT mode awaiting a command as per the previous startup instructions.

1.6.1 V6 Firmware Version

The location of the V6 firmware version is in offsets 003Eh (= F005) and 003Fh (= 00**xx**, where **xx** is a hex number corresponding to the alphabet numerical order).

A: Revisions

The table in this appendix gives a brief summary of any technical revisions made to this manual. When reading this manual online, you can jump to the first citation of a revision by clicking the links in blue.

Note: Only technical revisions appear in the table. Most even numbered pages contain a date stamp in the footer. If the footer date is more recent then the latest revision date given in the table, then the newest revision of this manual contains only *non-technical* revisions.

Revision Number	Revision Date	Description
2	12 Jul 2004	Added this Revisions chapter. This revision also incorporates all previous revisions.
2.01	9 Jan 2006	Corrected note in A24 Address Mode side text to read “To enable the Autoboot feature, set SW2 position 10, to the “off” position.”
2.02	13 Mar 06	Moved note about setting Autoboot feature to end of Table 9.2.4.